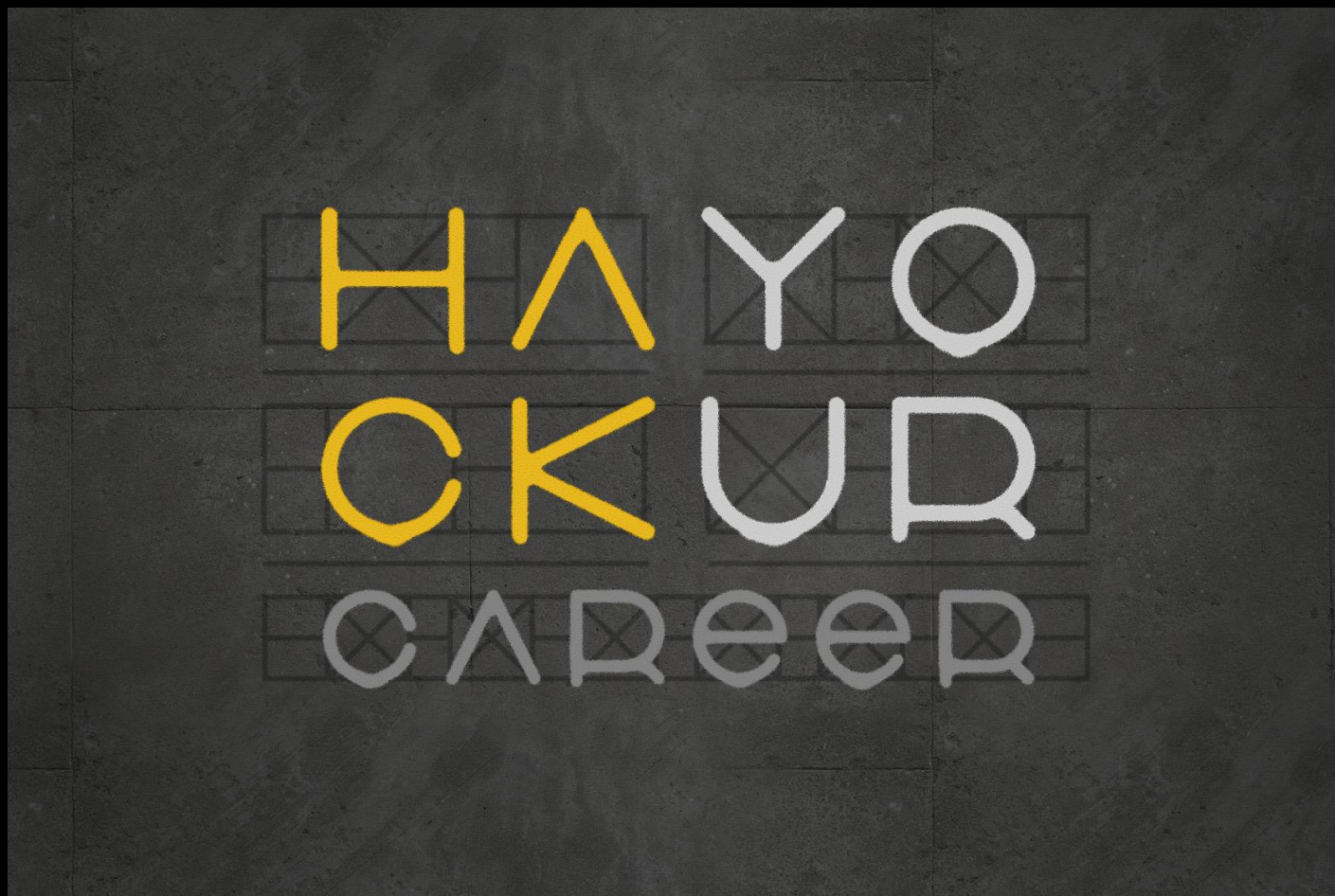


SAP Customer Experience

Spotkanie ML z Chmurą i CI/CD

Mateusz Klimas, SAP
Kwiecień 27, 2021

PUBLIC



Agenda

Support

- Czyli czym tak naprawdę się zajmujemy i po co nam kategoryzacja

Machine Learning

- Czym jest ML, jakie są jego rodzaje i gdzie można go zastosować

Implementacja projektu

- Bliższy rzut oka na wykorzystane technologie oraz jak powstał projekt

Wielki crossover

- Czyli jak chmura oraz CI/CD pomogły w projekcie

Demo

- Prezentacja projektu – czyli jak każdy może stworzyć swój model
- Stworzymy aplikację rozpoznającą czy wiadomość jest spamem z wykorzystaniem CI/CD

Kahoot Quiz oraz Q&A

Hack Your Career

HAYO
CKUR
career

HAYO
CKUR
career

02.03
2021

KEN LOMAX

CLOUD LAB 1: TAKING A SIMPLE GO APP THROUGH DOCKER AND KUBERNETES TO GOOGLE CLOUD

23.03
2021

JAN MĘDREK, MICHAŁ DRZEWIECKI

KICKSTART YOUR CAREER - JAK DZIAŁA PROGRAM STAŻOWY TEAM HACK W SAP LABS POLAND

13.04
2021

KAMIL SPUTO

TWORZENIE APLIKACJI Z WYKORZYSTANIEM GOOGLE CLOUD PLATFORM

27.04
2021

MATEUSZ KLIMAS

SPOTKANIE MACHINE LEARNING'U Z CHMURĄ I CI/CD

11.05
2021

PRZEMYSŁAW GOLICZ

WPROWADZENIE DO SERVICE MESH NA PRZYKŁADZIE ISTIO

18.05
2021

KATARZYNA PETKA, PATRYK MUSIOL

WYSŁANIE WEBSOCKETOWEJ APLIKACJI JENKINSEM W CHMURY

25.05
2021

WOJCIECH WRZALIK

PRACA Z PLATFORMĄ KUBERNETES

SAP

SAP Labs Poland

<https://hackyourcareer.github.io/>
<https://www.facebook.com/Hack.your.Career/>

SAP Labs Poland

Różnorodność domen

E-commerce, social commerce, cloud platform, open source,...

Development: Go, Java, Angular, Cloud Native solutions, ...

Możliwość nauki i rozwoju:

Platformy do edukacji on-line, certyfikacje, łatwy dostęp do ekspertów na całym świecie



> **350** pracowników

Różnorodność specjalizacji

Product teams, project teams, innovation projects teams, support teams

Jedno z 21 centrów
SAP's Labs Network

Kim jestem?

Mateusz Klimas

- Cloud Support Engineer
- Student Politechniki Śląskiej w Gliwicach – Informatics



Po co **kategoryzacja**?

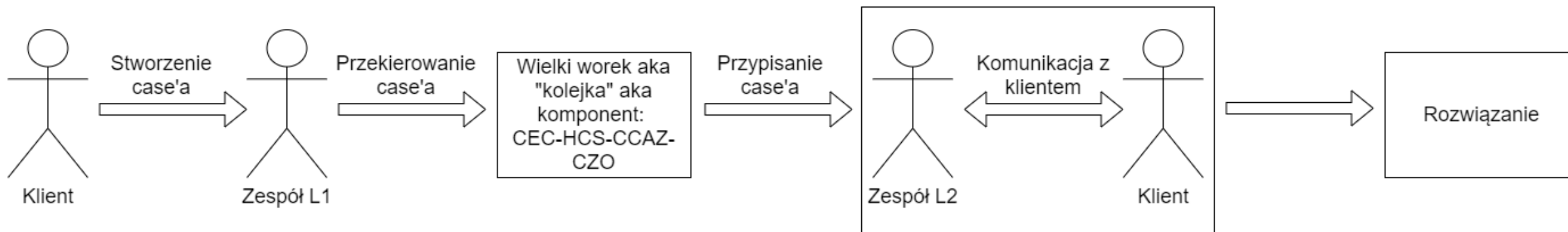


Co tak na prawdę robi product support?

- Wspieramy platformę e-commerce
- Rozwiązujemy techniczne problemy klientów z produktem
- Dostarczamy RCA (Root Cause Analysis)
- Wykonujemy operacje na środowiskach klienckich
- Dostarczamy informacji o produkcie



Żywot „kejsa”












Cel?

- Zamienić wielki worek na mniejsze worki

Co nam to da?

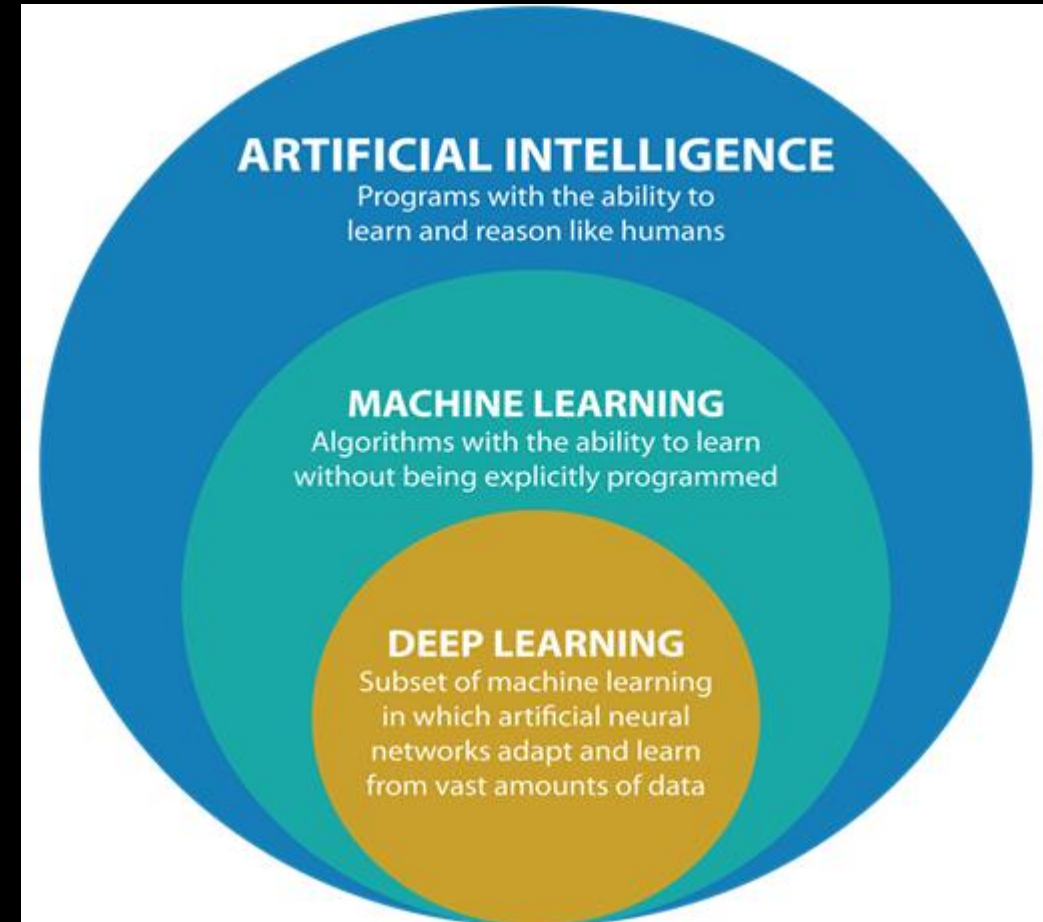
- Możliwość zdefiniowania najbardziej powszechnych problemów
- Lepsze zrozumienie gdzie brakuje nam wiedzy
- Pomoże przypisywaniu się do case'ów inżynierom specjalizujących konkretnym temacie lub chcących nauczyć się nowego
- Pomoże zarządzać kolejką (tym wielkim worem)

P2 > 2hrs  175679 Smartedit not working for |
P2 > 2hrs   259966 product syncing from ECC |
P2 > 1hr 364562 Media conversion failure |
P2 > 1hr  365061 Backoffice slowness / Que |
P2 < 1hr  365377 SOLR NAT endpoint redirec |
P2 < 1hr   268411 impex export does not wor |
P2 < 1hr 365318 Business process get stuc |
P2 < 1hr  264644 Kubernetes cluster versio |
P2 < 1hr  365141 HAC database DTU monitori |

Machine Learning

Machine Learning vs Artificial Intelligence vs Deep Learning

- Artificial Intelligence:
 - Smart Asystenci
 - Roboty produkcyjne
- Machine Learning:
 - Algorytmy klasyfikacyjne, predykcyjne
 - Diagnozy medyczne
- Deep Learning:
 - Sieci Neuronowe
 - Rozpoznawanie obrazów
 - Autonomiczne samochody



Czym jest dataset?

Dataset, czyli zestaw danych składa się w najczęściej z dwóch komponentów: wierszów i kolumn. Kluczową cechą zestawu danych jest to, że jest on zorganizowany w taki sposób, aby każdy wiersz zawierał jedną obserwację.

Dataset

Name	Year	Gender	NAPLAN Average
James	7	M	732
Nina	9	F	586
Eleanor	7	F	612
Joe	5	M	517
Melanie	7	M	716

Data items (variables)

Data unit

Observations

Supervised vs Unsupervised Learning

Supervised Learning

Input & Output Data

- Classification
- Regression

Predictions & Predictive
Models

Unsupervised Learning

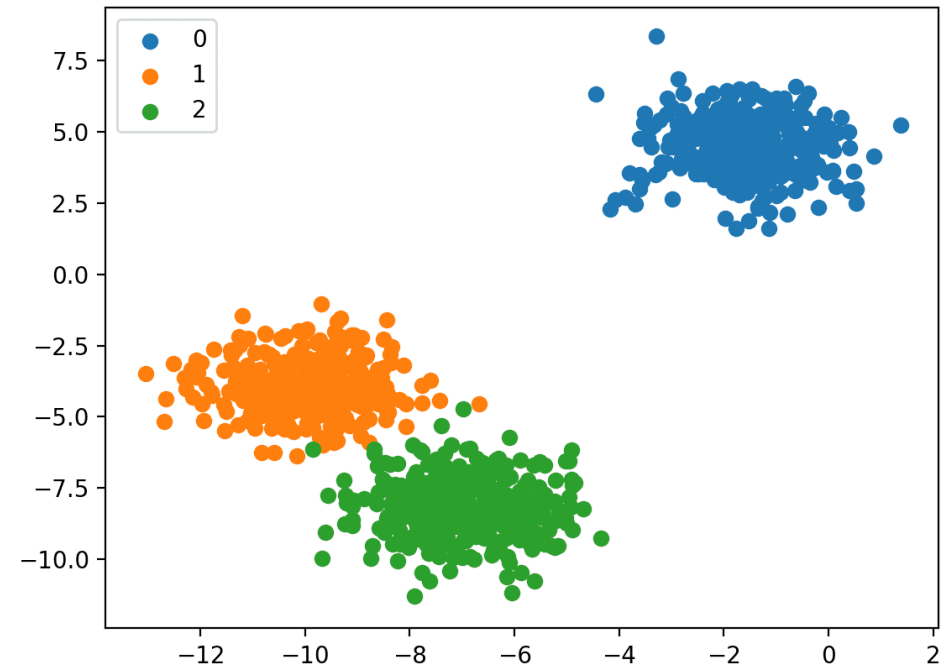
Input Data

- Clustering
- Association

Pattern / Structure
Discovery

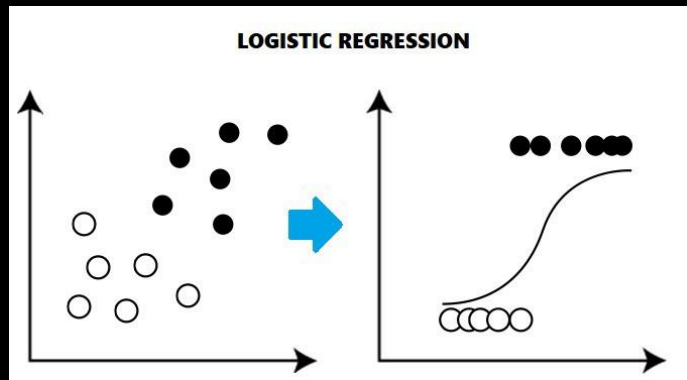
Problem klasyfikacji

W zadaniu klasyfikacyjnym dane wyjściowe algorytmu należą do jednej z różnych wstępnie wybranych kategorii. Model klasyfikacji próbuje przewidzieć wartość wyjściową, gdy ma kilka zmiennych wejściowych, umieszczając przykład we właściwej kategorii. Przykładem jest klasyfikowanie wiadomości e-mail jako „spam” lub „nie spam”.

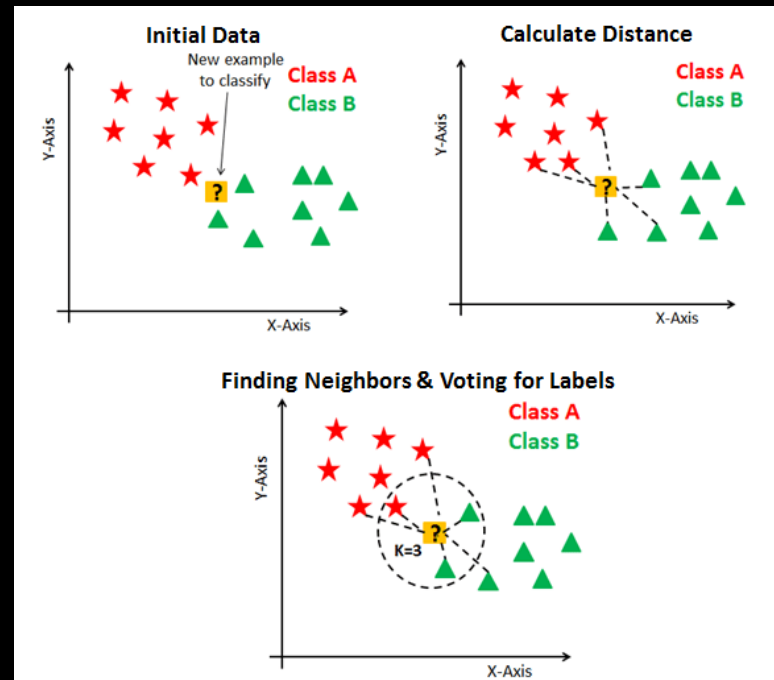


Algorytmy klasyfikacyjne

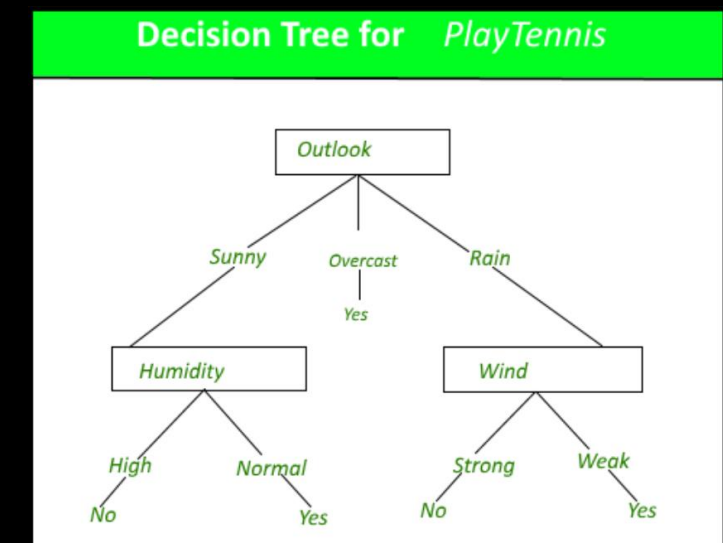
Regresja Logistyczna



K-Nearest Neighbours



Decision Tree

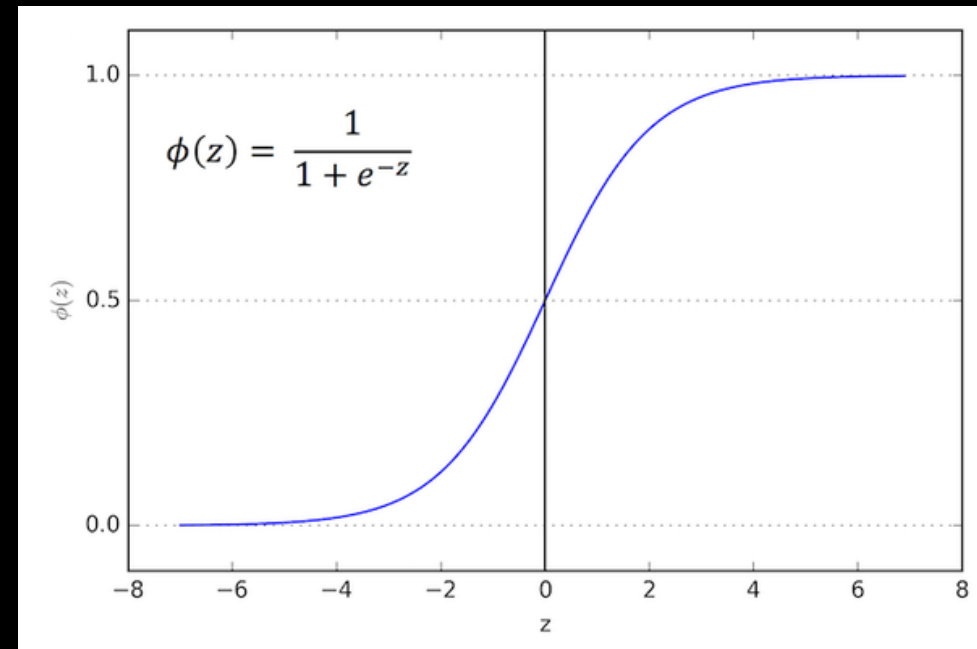
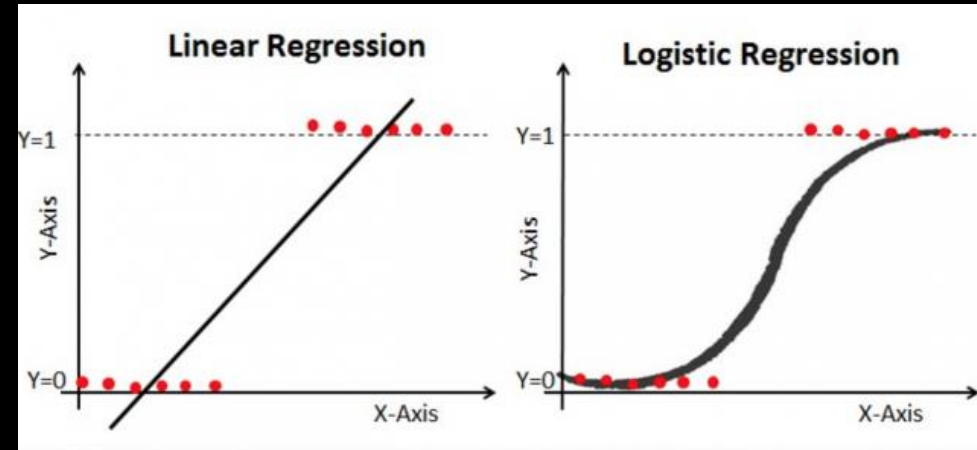


Logistic Regression

Celem regresji liniowej jest znalezienie linii, do której odległość od wszystkich punktów jest najmniejsza.

Regresja logistyczna mierzy związek między zmienną zależną, a co najmniej jedną zmienną niezależną, szacując prawdopodobieństwa za pomocą funkcji logistycznej.

Wynikiem regresji liniowej jest liczba (integer lub float) natomiast regresji logistycznej wartość z przedziału [0;1].



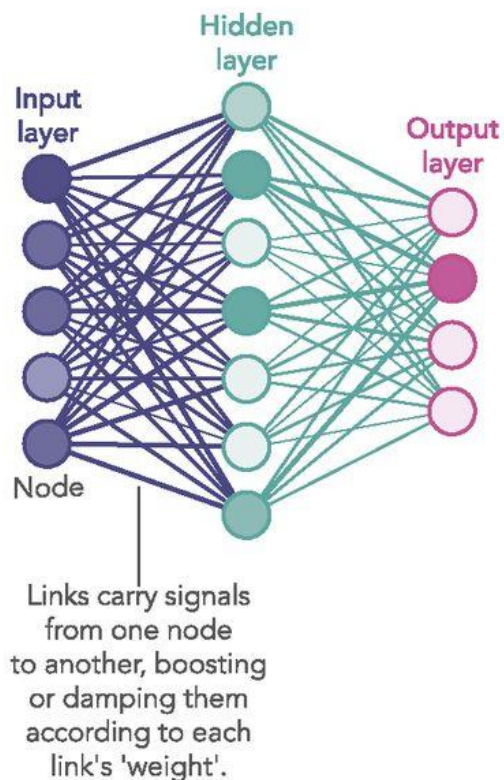


Deep Learning

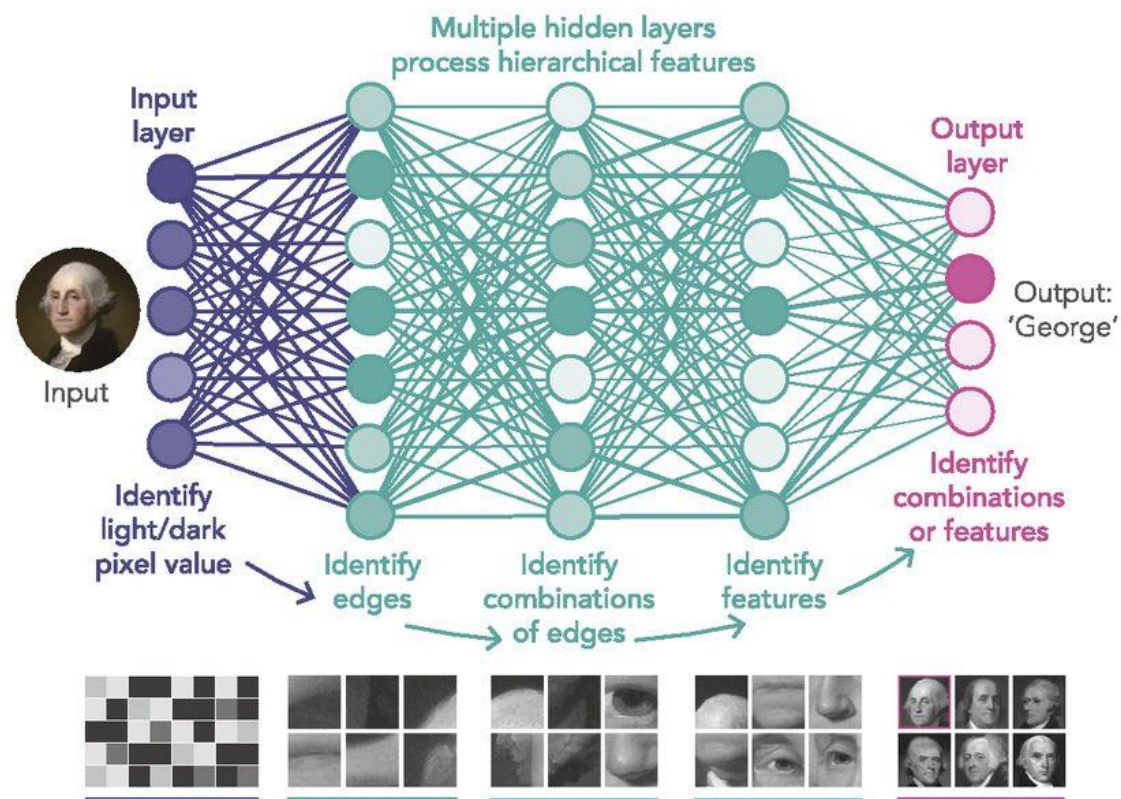
Deep learning jest uniwersalny i nadaje się do bardziej skomplikowanych danych:

- Generowanie obrazów
- Generowanie muzyki
- Klasyfikacja
- I wiele innych

1980S-ERA NEURAL NETWORK



DEEP LEARNING NEURAL NETWORK



Component Suggester

Proof of concept

Czyli w skrócie, dowód na to, że cały projekt ma sens.

Pierwszy prototyp składał się z:

- Bardzo prostego programu wykonywanego z wiersza poleceń
- Datasetu złożonego z ~200 rekordów zawierających tytuł problemu oraz etykietę
- Modelu rozpoznającego ~8 klas

Ale jak to nie mamy datasetu?

Próba stworzenia datasetu

- Przejrzenie danych jakie mamy
- Zaangażowanie zespołu, by pomógł kategoryzować case'y

Efekt?

- Około 200 rekordów po 2tygodniach

Problem z danymi

- Ogromny „szum informacyjny”

Ja: Czy mamy dataset?

Manager: Nope

Ja:



Product Area selected: SAP Commerce Cloud
Product Function selected: SAP Commerce Cloud > SAP Commerce Cloud
Component selected (Manual Selection): CEC-HCS

--- Description ---
Hi!

We can no longer synchronize our catalogs. For some reason, it gets stuck and we cannot do anything to abort and restart it.

The relevant threads are:

```
"Thread-217" prio=5 tid=0x582 nid=0x246 RUNNABLE (JNI Native Code) - stats: cpu=52 blk=0 wait=7 java.lang.Thread.State: RUNNABLE at java.base@11.0.7/java.net.SocketInputStream.socketRead0(Native Method) at java.base@11.0.7/java.net.SocketInputStream.socketRead(SocketInputStream.java:115) at java.base@11.0.7/java.net.SocketInputStream.read(SocketInputStream.java:140) at com.microsoft.sqlserver.jdbc.TDSChannel$ProxyInputStream.readInternal(IOBuffer.java:1006) at com.microsoft.sqlserver.jdbc.TDSChannel$ProxyInputStream.read(IOBuffer.java:996) at java.base@11.0.7/sun.security.ssl.SSLSocketImpl$AppInputStream.read(SSLSocketImpl$AppInputStream@6538e7b8 at com.microsoft.sqlserver.jdbc.TDSChannel.read(IOBuffer.java:2026) at com.microsoft.sqlserver.jdbc.TDSReader.readPacket(IOBuffer.java:6446) - locked com.microsoft.sqlserver.jdbc.TDSReader@4f099bb7 at com.microsoft.sqlserver.jdbc.TDSCommand.startResponse(IOBuffer.java:7610) at com.microsoft.sqlserver.jdbc.SQLServerPreparedStatement.doExecutePreparedStatement(SQLServerPreparedStatement.java:598) at com.microsoft.sqlserver.jdbc.SQLServerPreparedStatement$PrepStmtExecCmd.doExecute(SQLServerPreparedStatement.java:522) at com.microsoft.sqlserver.jdbc.TDSCommand.execute(IOBuffer.java:7225) at com.microsoft.sqlserver.jdbc.SQLServerConnection.executeCommand(SQLServerConnection.java:3053) - locked java.lang.Object@80dcb1 at com.microsoft.sqlserver.jdbc.SQLServerStatement.executeCommand(SQLServerStatement.java:247) at com.microsoft.sqlserver.jdbc.SQLServerStatement.executeStatement(SQLServerStatement.java:222) at com.microsoft.sqlserver.jdbc.SQLServerPreparedStatement.executeQuery(SQLServerPreparedStatement.java:444) at de.hybris.platform.jdbcwrapper.interceptor.PreparedStatementWithJDBCInterceptor.lambda$3(PreparedStatementWithJDBCInterceptor.java:76) at de.hybris.platform.jdbcwrapper.interceptor.PreparedStatementWithJDBCInterceptor$$Lambda$184/0x00000000800551c40.get(Unknown Source) at de.hybris.platform.jdbcwrapper.interceptor.recover.SQLRecoverableExceptionHandler.passThrough(SQLRecoverableExceptionHandler.java:101) at de.hybris.platform.jdbcwrapper.interceptor.recover.SQLRecoverableExceptionHandler.get(SQLRecoverableExceptionHandler.java:59) at de.hybris.platform.jdbcwrapper.interceptor.JDBCInterceptor.get(JDBCInterceptor.java:69) at de.hybris.platform.jdbcwrapper.interceptor.PreparedStatementWithJDBCInterceptor.executeQuery(PreparedStatementWithJDBCInterceptor.java:76) at de.hybris.platform.jdbcwrapper.PreparedStatementImpl.executeQuery(PreparedStatementImpl.java:196) at de.hybris.platform.catalog.jalo.synchronization.JDBCQuery.execute(JDBCQuery.java:67) at de.hybris.platform.catalog.jalo.synchronization.SyncSchedulerCallableBase.callImpl(SyncSchedulerCallableBase.java:122) at de.hybris.platform.catalog.jalo.synchronization.SyncSchedulerCallableBase.call(SyncSchedulerCallableBase.java:96) at de.hybris.platform.catalog.jalo.synchronization.SyncSchedulerCallableBase.call(SyncSchedulerCallableBase.java:1) at java.base@11.0.7/java.util.concurrent.FutureTask.run(FutureTask.java:264) at java.base@11.0.7/java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1128) at java.base@11.0.7/java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:628) at de.hybris.platform.core.TenantAwareThreadFactory$1.internalRun(TenantAwareThreadFactory.java:159) at de.hybris.platform.core.threadregistry.RegistrableThread.run(RegistrableThread.java:131) Locked synchronizers: count = 1 - java.util.concurrent.ThreadPoolExecutor$Worker@1c7cfee5
```

```
"0001PN4W::de.hybris.platform.catalog.jalo.synchronization.CatalogVersionSyncJob" prio=5 tid=0xc8 nid=0xc8 WAITING - stats: cpu=2064 blk=38 wait=1691269 java.lang.Thread.State: WAITING at java.base@11.0.7/jdk.internal.misc.Unsafe.park(Native Method) at java.base@11.0.7/java.util.concurrent.locks.LockSupport.park(LockSupport.java:194) at java.base@11.0.7/java.util.concurrent.FutureTask.awaitDone(FutureTask.java:447) at java.base@11.0.7/java.util.concurrent.FutureTask.get(FutureTask.java:190) at java.base@11.0.7/java.util.concurrent.AbstractExecutorService.invokeAll(AbstractExecutorService.java:247) at de.hybris.platform.catalog.jalo.synchronization.CatalogVersionSyncJob.scheduleCreationOfNewItems(CatalogVersionSyncJob.java:668) at de.hybris.platform.catalog.jalo.synchronization.CatalogVersionSyncJob.configureFullVersionSync(CatalogVersionSyncJob.java:473) at de.hybris.platform.catalog.jalo.synchronization.CatalogVersionSyncJob.performCronJob(CatalogVersionSyncJob.java:846) at de.hybris.platform.cronjob.jalo.Job.execute(Job.java:1401) at de.hybris.platform.cronjob.jalo.Job.performImpl(Job.java:812) at de.hybris.platform.cronjob.jalo.Job$JobRunnable.run(Job.java:681) at de.hybris.platform.util.threadpool.PoolableThread.internalRun(PoolableThread.java:206) at de.hybris.platform.core.threadregistry.RegistrableThread.run(RegistrableThread.java:131) Locked synchronizers: count = 0
```

I have attached the full thread dump.

In essence, the database does not respond and therefore the synchronization cannot proceed.

This is probably related to ticket 818912/2020 and possibly also 773649/2020.

A może by tak generować dane?

Każdy problem posiada swoje „słowa klucze”

Po wielu godzinach testów najlepsze rezultaty uzyskaliśmy dla:

- Regresji Logistycznej
- Losowego łączenia słów kluczy w pary
<słowo-klucz> + <słowo-klucz>

Idea generycznego kodu:

- Możliwość manipulowania danymi bez ingerencji w kod

```
{
  "name": "CEC-HCS-CCAZ-OPS",
  "nouns": [
    "DTU",
    "thread dump",
    "heap dump",
    "redis"
  ],
  "adjectives": [
    "heap",
    "number of pods",
    "memory",
    "CPU",
    "disk",
    "RAM",
    "resources",
    "size",
    "slowness",
    "VPN",
    "VPN Tunnel",
    "replica",
    "failure rate increase",
    "load test",
    "restart"
  ],
  "templates": [
    " {{noun}} {{noun}} "
  ]
}
```

```
{
  "title": " heap dump CPU .",
  "category": "CEC-HCS-CCAZ-OPS",
  "description": ""
},
{
  "title": " load test RAM ? ",
  "category": "CEC-HCS-CCAZ-OPS",
  "description": ""
},
{
  "title": " thread dump failure rate increase",
  "category": "CEC-HCS-CCAZ-OPS",
  "description": ""
},
{
  "title": " redis failure rate increase ;",
  "category": "CEC-HCS-CCAZ-OPS",
  "description": ""
},
{
  "title": " DTU number of pods .",
  "category": "CEC-HCS-CCAZ-OPS",
  "description": ""
},
{
  "title": " load test CPU .",
  "category": "CEC-HCS-CCAZ-OPS",
  "description": ""
},
{
  "title": " heap dump thread dump .",
  "category": "CEC-HCS-CCAZ-OPS",
  "description": ""
},
}
```

Product Area selected: SAP Commerce Cloud
Product Function selected: SAP Commerce Cloud > SAP Commerce Cloud
Component selected (Manual Selection): CEC-HCS

--- Description ---
Hi!

We can no longer synchronize our catalogs. For some reason, it gets stuck and we cannot do anything to abort and restart it.

The relevant threads are:

```
"Thread-217" prio=5 tid=0x582 nid=0x246 RUNNABLE (JNI Native Code) - stats: cpu=52 blk=0 wait=7 java.lang.Thread.State: RUNNABLE at java.base@11.0.7/java.net.SocketInputStream.socketRead0(Native Method) at java.base@11.0.7/java.net.SocketInputStream.socketRead(SocketInputStream.java:115) at java.base@11.0.7/java.net.SocketInputStream.read(SocketInputStream.java:140) at com.microsoft.sqlserver.jdbc.TDSChannel$ProxyInputStream.readInternal(IOBuffer.java:1006) at com.microsoft.sqlserver.jdbc.TDSChannel$ProxyInputStream.read(IOBuffer.java:996) at java.base@11.0.7/sun.security.ssl.SSLSocketImpl$AppInputStream.read(SSLSocketImpl$AppInputStream@6538e7b8 at com.microsoft.sqlserver.jdbc.TDSChannel.read(IOBuffer.java:2026) at com.microsoft.sqlserver.jdbc.TDSReader.readPacket(IOBuffer.java:6446) - locked com.microsoft.sqlserver.jdbc.TDSReader@4f099bb7 at com.microsoft.sqlserver.jdbc.TDSCommand.startResponse(IOBuffer.java:7610) at com.microsoft.sqlserver.jdbc.SQLServerPreparedStatement.doExecutePreparedStatement(SQLServerPreparedStatement.java:598) at com.microsoft.sqlserver.jdbc.SQLServerPreparedStatement$PrepStmtExecCmd.doExecute(SQLServerPreparedStatement.java:522) at com.microsoft.sqlserver.jdbc.TDSCommand.execute(IOBuffer.java:7225) at com.microsoft.sqlserver.jdbc.SQLServerConnection.executeCommand(SQLServerConnection.java:3053) - locked java.lang.Object@80dcb1 at com.microsoft.sqlserver.jdbc.SQLServerStatement.executeCommand(SQLServerStatement.java:247) at com.microsoft.sqlserver.jdbc.SQLServerStatement.executeStatement(SQLServerStatement.java:222) at com.microsoft.sqlserver.jdbc.SQLServerPreparedStatement.executeQuery(SQLServerPreparedStatement.java:444) at de.hybris.platform.jdbcwrapper.interceptor.PreparedStatementWithJDBCInterceptor.lambda$3(PreparedStatementWithJDBCInterceptor.java:76) at de.hybris.platform.jdbcwrapper.interceptor.PreparedStatementWithJDBCInterceptor$$Lambda$184/0x0000000800551c40.get(Unknown Source) at de.hybris.platform.jdbcwrapper.interceptor.recover.SQLRecoverableExceptionHandler.passThrough(SQLRecoverableExceptionHandler.java:101) at de.hybris.platform.jdbcwrapper.interceptor.recover.SQLRecoverableExceptionHandler.get(SQLRecoverableExceptionHandler.java:59) at de.hybris.platform.jdbcwrapper.interceptor.JDBCInterceptor.get(JDBCInterceptor.java:69) at de.hybris.platform.jdbcwrapper.interceptor.PreparedStatementWithJDBCInterceptor.executeQuery(PreparedStatementWithJDBCInterceptor.java:76) at de.hybris.platform.jdbcwrapper.PreparedStatementImpl.executeQuery(PreparedStatementImpl.java:196) at de.hybris.platform.catalog.jalo.synchronization.JDBCQuery.execute(JDBCQuery.java:67) at de.hybris.platform.catalog.jalo.synchronization.SyncSchedulerCallableBase.callImpl(SyncSchedulerCallableBase.java:122) at de.hybris.platform.catalog.jalo.synchronization.SyncSchedulerCallableBase.call(SyncSchedulerCallableBase.java:96) at de.hybris.platform.catalog.jalo.synchronization.SyncSchedulerCallableBase.call(SyncSchedulerCallableBase.java:1) at java.base@11.0.7/java.util.concurrent.FutureTask.run(FutureTask.java:264) at java.base@11.0.7/java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1128) at java.base@11.0.7/java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:628) at de.hybris.platform.core.TenantAwareThreadFactory$1.internalRun(TenantAwareThreadFactory.java:159) at de.hybris.platform.core.threadregistry.RegistrableThread.run(RegistrableThread.java:131) Locked synchronizers: count = 1 - java.util.concurrent.ThreadPoolExecutor$Worker@1c7cfee5
```

```
"0001PN4W::de.hybris.platform.catalog.jalo.synchronization.CatalogVersionSyncJob" prio=5 tid=0xc8 nid=0xc8 WAITING - stats: cpu=2064 blk=38 wait=1691269 java.lang.Thread.State: WAITING at java.base@11.0.7/jdk.internal.misc.Unsafe.park(Native Method) at java.base@11.0.7/java.util.concurrent.locks.LockSupport.park(LockSupport.java:194) at java.base@11.0.7/java.util.concurrent.FutureTask.awaitDone(FutureTask.java:447) at java.base@11.0.7/java.util.concurrent.FutureTask.get(FutureTask.java:190) at java.base@11.0.7/java.util.concurrent.AbstractExecutorService.invokeAll(AbstractExecutorService.java:247) at de.hybris.platform.catalog.jalo.synchronization.CatalogVersionSyncJob.scheduleCreationOfNewItems(CatalogVersionSyncJob.java:668) at de.hybris.platform.catalog.jalo.synchronization.CatalogVersionSyncJob.configureFullVersionSync(CatalogVersionSyncJob.java:473) at de.hybris.platform.catalog.jalo.synchronization.CatalogVersionSyncJob.performCronJob(CatalogVersionSyncJob.java:846) at de.hybris.platform.cronjob.jalo.Job.execute(Job.java:1401) at de.hybris.platform.cronjob.jalo.Job.performImpl(Job.java:812) at de.hybris.platform.cronjob.jalo.Job$JobRunnable.run(Job.java:681) at de.hybris.platform.util.threadpool.PoolableThread.internalRun(PoolableThread.java:206) at de.hybris.platform.core.threadregistry.RegistrableThread.run(RegistrableThread.java:131) Locked synchronizers: count = 0
```

I have attached the full thread dump.

In essence, the database does not respond and therefore the synchronization cannot proceed.

This is probably related to ticket 818912/2020 and possibly also 773649/2020.

Mając „w miare” działający model należało się zastanowić jak dostarczyć go do użytkownika.

W tym momencie projekt podzielił się na trzy części

- Client
 - Część odpowiedzialna za wykonywanie skryptu po stronie użytkownika
- Server
 - Część odpowiedzialna za to co dzieje się po stronie serwera
- Model
 - Część odpowiadająca za szkolenie modelu

Rzut okiem na technologie

Model:

- Logistic Regression
- Node.js
 - Open-source, między platformowe, back-endowe środowisko uruchomieniowe pozwalające wykonywać kod JavaScript poza przeglądarką
- Natural.js
 - Biblioteka zawierająca implementacje algorytmu regresji logistycznej



Rzut okiem na technologie

Client:

- ViolentMonkey
 - manager skryptów open-source, pozwala na automatyczne wykonywanie i zarządzanie skryptami w przeglądarce
- JavaScript
 - Kto pisze w JSie ten się w cyrku nie śmieje

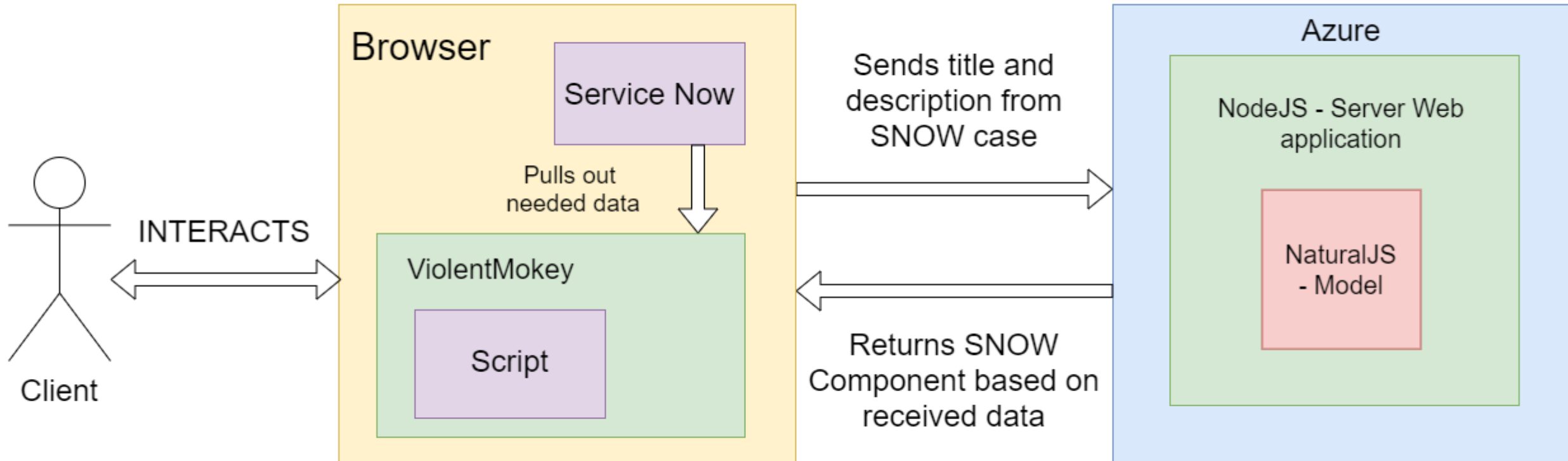


Rzut okiem na technologie

Serwer:

- Microsoft Azure
 - Usługa chmurowa stworzona dla budowania, testowania, wdrażania i zarządzania serwisami poprzez centra danych Microsoftu / dostawca chmury, gdzie „stoi” nasza aplikacja jak i Jenkins.
- Express.js
 - Back-endowy framework dla node.js pozwalający tworzyć aplikacje webowe oraz API. W praktyce standardowy framework serwerowy dla node.js.





Pierwszy „pipeline”

„Ludzki Pipeline” składał się z:

- Lokalnie u siebie generowałem model – często parę godzin
- Po wygenerowaniu modelu archiwizowałem go i wysyłałem na Slacku do Leszka
- Leszek ręcznie aktualizował model na serwerze
- Zostawał ręcznie włączony „deployment” poprzez komunikację z API Azura

Wielki Crossover

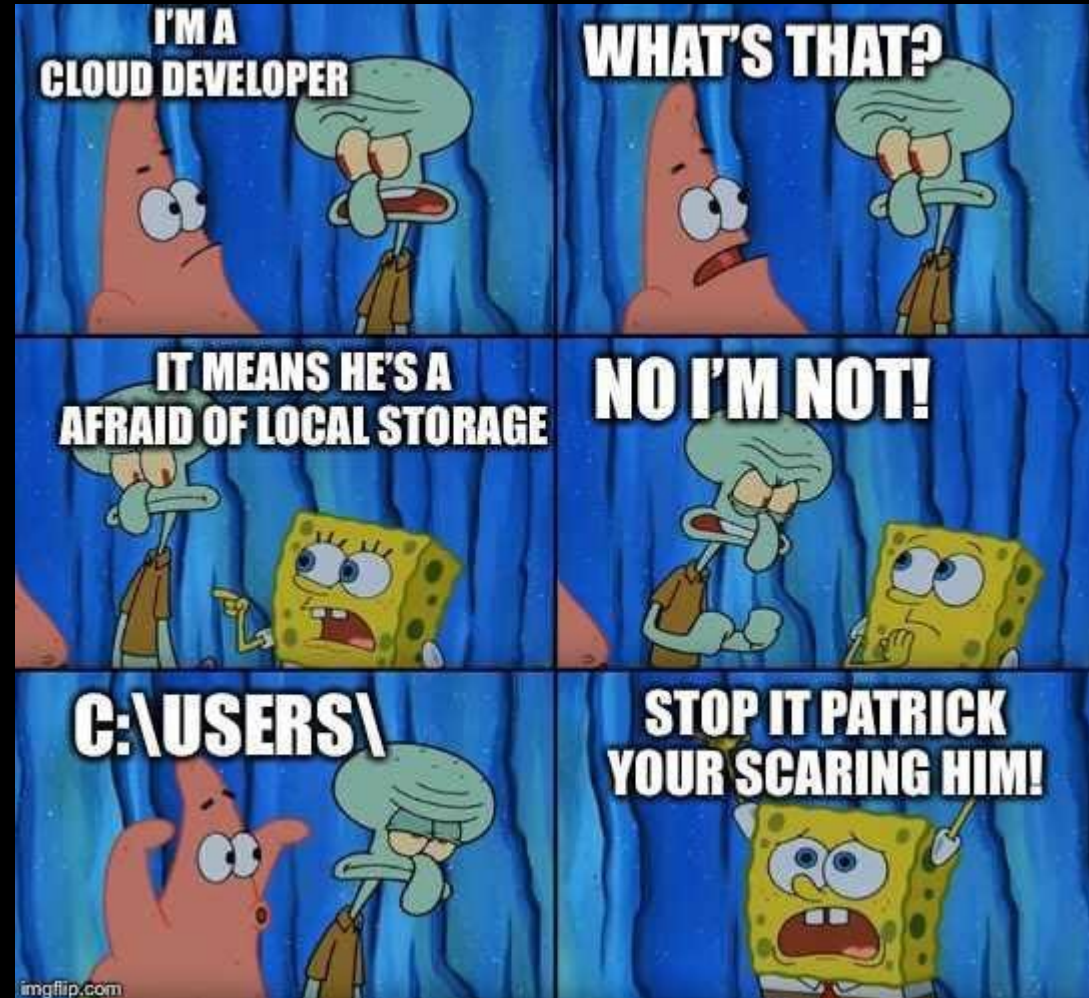
Czym w sumie jest chmura?

Chmurka jest:

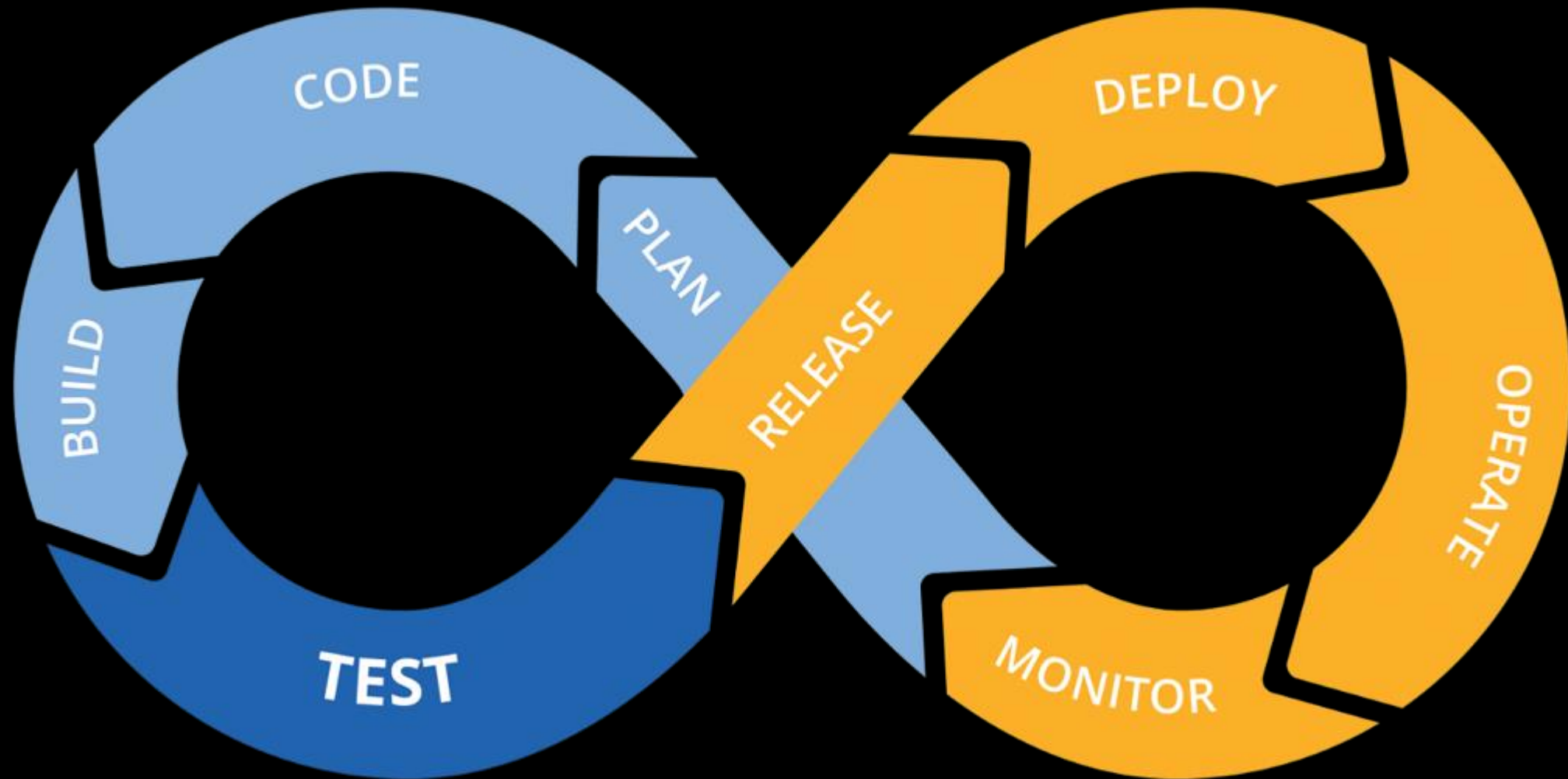
- Serwerami
- Zajętą przestrzenią
- Zużytych zasobami
- Połączeniem z Internetem
- Cudem na Ziemi

Rodzaje zasobów w chmurze:

- Dyski
- Maszyny wirtualne
- Bazy danych
- Scale sety
- Load balancery
- i wiele więcej...



Czym w sumie jest CI/CD aka Continuous Integration, Continuous Delivery?



Dwa środowiska

Stg (Staging)

- Środowisko wykorzystywane do celów testowych, eksperymentacyjnych. Głównie połączone z procesem rozwijania nowych ficzerów, poprawiania modelu czy pipeline'ów etc.

Prod (Production)

- Środowisko odpowiedzialne za obsługę żądań wysyłanych przez użytkowników końcowych. Jest monitorowane i utrzymywane, by zapewnić jak najlepsze doświadczenia użytkownikom. Tylko „działające” i „przetestowane” wersje są tam wdrażane.

Automatyzacja - Jenkins

Jenkins jest darmowym oraz open-source serwerem pozwalającym na automatyzację. Pomaga zautomatyzować takie procesy jak: budowanie, testowanie czy też wdrażanie projektu.



A w praktyce?

- Instalujemy Jenkinsa (nieważne czy lokalnie czy na zewnętrznych maszynach)
- Opcjonalnie instalujemy pluginy
 - Rozszerzają one możliwości Jenkinsa np. Integracja z github lub MS Azure
- Tworzymy pipeline
- Naciskamy guzik i patrzymy jak dzieją się czary



Co można zautomatyzować?

Dla przypomnienia jak to działało:

- Lokalnie u siebie generowałem model – często parę godzin
- Po wygenerowaniu modelu archiwizowałem go i wysyłałem na Slacku do Leszka
- Leszek ręcznie aktualizował model na serwerze
- Zostawał ręcznie włączony „deployment” poprzez komunikację z API Azura

Pipeline'y w projekcie

- classifai-tool-pipeline – odpowiedzialny za deployment na środowisko stg
- classifai-tool-prod-pipeline – odpowiedzialny za deployment na środowisko prod
- classifai-tool-release-pipeline – odpowiedzialny za release na githubie
- train-model – odpowiedzialny za szkolenie modelu
- model-check – odpowiedzialny za przetestowanie modelu

Pipeline model-check

Full project name: CI/model-check



Last Successful Artifacts

results.json

69.97 KB



view



Recent Changes

Stage View



Małe podsumowanie

- Zaczęliśmy z malutkim projektem bez datasetu
- Generujemy dane sami na podstawie słów kluczy i generycznego kodu
- Za pomocą MS Azure i ViolentMokey dostarczyliśmy produkt do użytkownika
- Za pomocą Jenkinsa w chmurze zautomatyzowane zostały procesy szkolenia oraz wdrażania

„Live” Demo 1



Panie programisto proszę mi dodać komponent; Sam se Pan dodaj!

Wcielimy się w rolę osoby „nietechnicznej”, która:

- Chce by model rozpoznawał nowe komponenty, czyli dodać nową etykietę

W tym celu będziemy musieli:

- Znaleźć słowa kluczowe dla problemu
- Wyszkolić nowy model
- Przetestować go
- Wdrożyć na środowisko

Live Demo 2



Czy to spam?

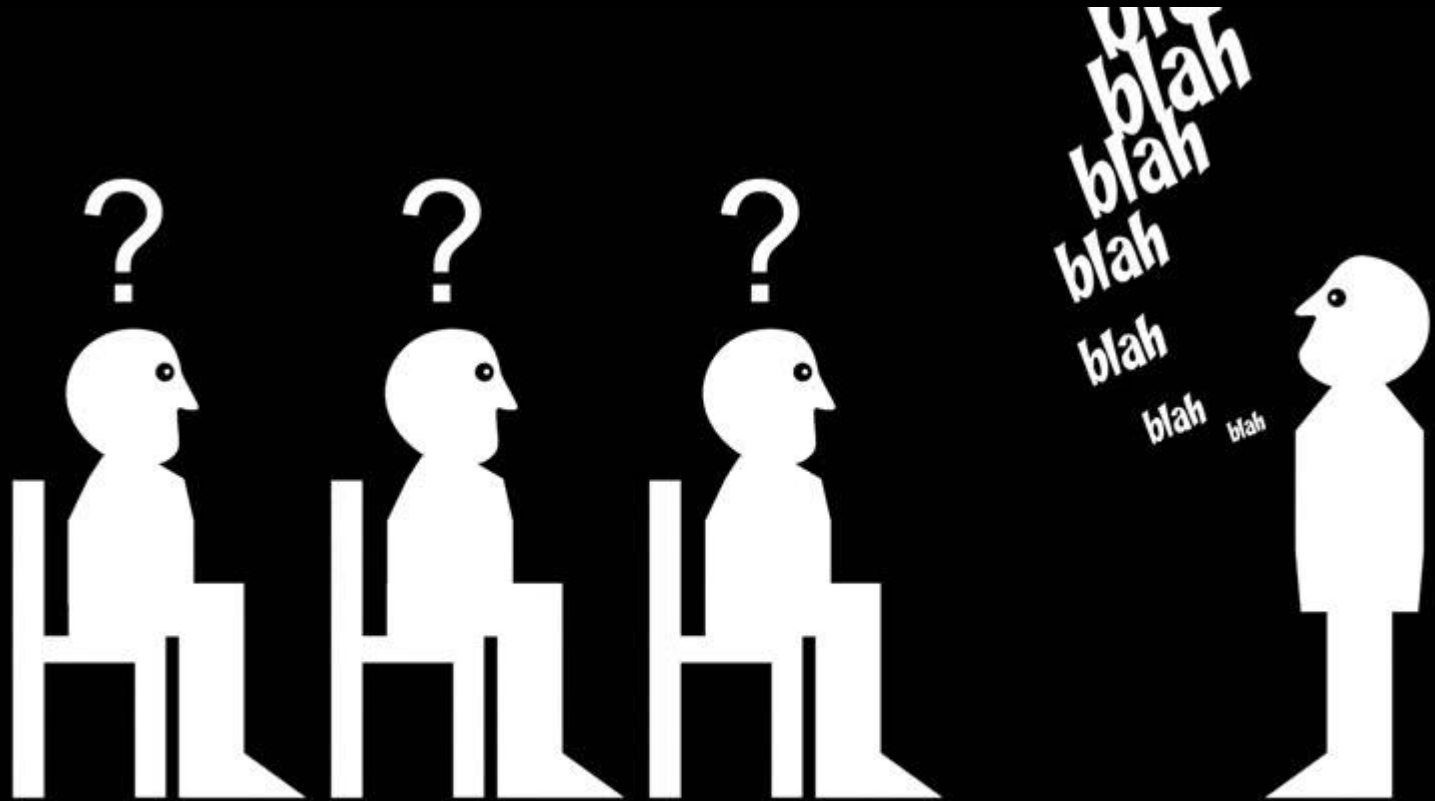
Stworzymy bardzo prostą aplikację wykorzystującą ML do detekcji czy dana wiadomość jest spamem czy też nie. Następnie zautomatyzujemy proces jej budowania i tworzenia modelu.

W tym celu będziemy musieli:

- Napisać aplikację (obviously)
- Umieścić ją w repozytorium
- Napisać pipeline Jenkinsowy
- A reszta zrobi za nas automatyzacja

Kahoot Quiz

Q&A
???



Przydatne materiały

Spam detection article

- <https://towardsdatascience.com/spam-detection-with-logistic-regression-23e3709e522>

Instalacja Jenkinsa

- <https://www.jenkins.io/doc/book/installing/>

Natural.js

- <https://github.com/NaturalNode/natural>

Wstęp do MS Azure

- <https://azure.microsoft.com/en-us/get-started/>

AI vs ML vs DL:

- <https://www.linkedin.com/pulse/what-artificial-intelligence-without-machine-learning-claudia-pohlink/>

Repozytorium do projektu:

- <https://github.com/Withel/hyc-ml-cicd>

Aktualne oferty pracy:

- <https://jobs.sap.com/search/?locationsearch=gliwice>
(<https://url.sap/3vywdy>)

Śledź nas na:

- <https://hackyourcareer.github.io/>
- <https://www.facebook.com/Hack.your.Career/>

Thank you.

Kontakt:

Mateusz Klimas

mateusz.klimas@sap.com

<https://www.linkedin.com/in/mat-klimas/>

