Eindrapport Data-Engineering-Project 2 | Groep 2

STUDENTEN:

- *Floris Buyse*
- *Emma De Koster*
- *Marlon Engels*
- *Max Milan*
- *Storm Tuyls*

BEGELEIDERS:

- *Johan Decorte*
- *Giselle Vercauteren*

Inhoudstafel

Epic 1

Korte uitleg van de epic

Het doel van deze epic was dat een keyuser de aangeleverde vokaledendata volledig kan consulteren in de omgeving van HoGent, en dat deze ad hoc queries kan uitvoeren om data kwaliteit te valideren.

Hiervoor hebben we een relationele databank opgezet waarin de aangeleverde data wordt bijgehouden. Deze data hebben we eerst gecleaned en enkel de data van Oost-Vlaanderen behouden. Vervolgens hebben we de data in een relationele databank gestoken. Deze databank dient dus voor de toestroom aan data. Daarna hebben we een Data Warehouse opgesteld die meer geschikt is voor het ophalen en analyseren van data. Tot slot hebben we ook een User Interface gemaakt (waar de data warehouse aan gelinkt is), a.d.h.v Streamlit waarop bepaalde queries kunnen worden uitgevoerd. Deze queries kunnen zelf geschreven worden en er zijn ook enkele voorgemaakte queries beschikbaar.

Beperkingen en uitdagingen

Het cleanen van de data was een lastige taak. Er was veel data die niet bruikbaar was (NaN waarden) en er waren ook veel verschillende formaten. Er zat weinig structuur in de data en die hebben we zelf moeten toevoegen wat veel tijd in beslag nam. Ook waren er veel problemen met de Foreign Keys en Primary Keys (niet bestaande primary key waarvan er wel een foreign key was bijvoorbeeld). De data types klopten ook niet altijd en moesten we vaak zelf aanpassen (zo kregen we de doorgestuurde datums bijvoorbeeld als strings idp datetime-objecten).

Daarnaast was het ook een uitdaging om de juiste layout voor de Data Warehouse te vinden. We hebben deze een aantal keer opnieuw moeten maken. Uiteindelijk zij we geland op een DWH met 1 facttable (nl FactInschrijving), dit omdat inschrijvingen eigenlijk hetgene is dat Voka 'verkoopt', dit is het product die ze eigenlijk aanbieden aan de bedrijven die ingeschreven zijn bij hun. Ook hebben we een deel data laten vallen en anders gestructureerd hier. Dit omdat niet alle oorspronkelijke data nuttig was voor de analyses/epics die wilden maken. Tot slot was het ook een uitdaging om de DWH op het VIC te krijgen. Eerst hadden we namelijk de pipeline lokaal opgezet let Integration Services in Visual Studio. Op het VIC hebben we dit dan opgelost met Views.

Bepaalde keuzes door beperkingen

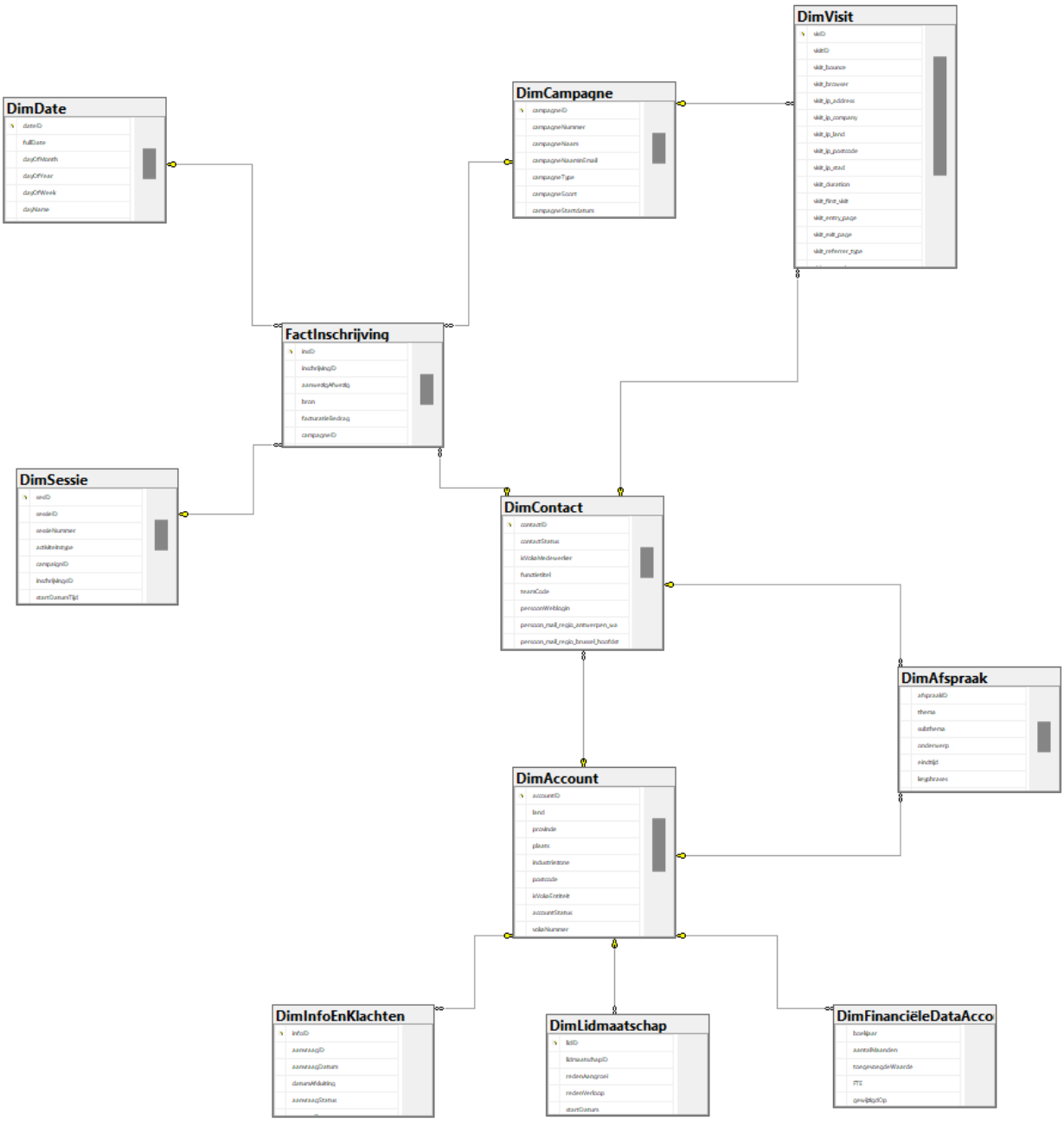
Soms hebben we enkele rijen data moeten laten vallen aangezien er hier weinig tot geen nuttige informatie in stond (veel NaN waarden). Dit heeft misschien tot gevolg dat er sommige bedrijven die wel in de data zitten niet meer in de uiteindelijke databank zitten.

Gedachtengang / Hoe zijn we tot de oplossingen gekomen

Eerst en vooral dienden heel wat algemene cleanups te gebeuren, zoals het verwijderen van / opvullen van NaN waarden. Ook hebben we de kolomnamen aangepast zodat deze meer leesbaar waren en allemaal hetzelfde formaat hebben. Daarnaast hebben we rijen moeten verwijderen die data hadden met Foreign Keys die verwezen naar niet-bestaande Primary Keys om geen problemen te krijgen met de databank. Als laatste hebben we ook nog de juiste datatypes moeten toekennen aan sommige kolommen (Bijvoorbeeld Float-getallen omgezet naar Integers en datums die als String in de data zaten, omgezet naar datatime-objecten).

We hebben de databank opgesteld aan de hand van het meegegeven ERD. Deze hebben we grondig geanalyseerd alvorens het opstellen van de databank. Vervolgens hebben we ons eigen ERD opgesteld, aangepast aan onze doorgevoerde wijzigingen. Dan hebben we met behulp van een Object Relational Mapper (ORM) de data in de database gestoken.

De data warehouse werd dan opgesteld aan de hand van een SQL-Server script. Het nodige Sterschema werd hier ook bij ontwikkeld. Tot slot wordt de data warehouse gevuld met behulp van views voor elke tabel van de DWH. De views halen de juiste data uit de relationele databank en steken deze in de DWH.



Epic 2

`Korte uitleg van de epic`

Het doel van epic 2 was het mogelijk maken om als keyuser nieuwe data (volgens hetzelfde model) vlot toe te voegen . Dit gaat typisch over de recentste gedragsdata, transactionele data en veranderingen in master data.

Wij hebben dit mogelijk gemaakt door middel van een gegeven CSV bestand. Dit CSV bestand wordt dan ingelezen en de data wordt toegevoegd aan de databank.

Beperkingen en uitdagingen

Het inlezen van de CSV bestanden was niet zozeer een uitdaging. Waar we wel wat moeilijkheden ervaren hebben, was bij het zorgen dat deze bestanden telkens dezelfde naam hebben. Ook is er mogelijks een probleem dat er fouten in de data zitten die niet opgemerkt worden, zoals verkeerde separators. Er waren ook problemen bij het toevoegen van reeds bestaande data, dit met foreign keys en primary keys.

Bepaalde keuzes door beperkingen

Doordat er problemen waren met de foreign keys en primary keys hebben we besloten om de data die al in de databank zit niet te overschrijven. Dit om geen problemen te krijgen met de databank. Hierdoor werkt deze epic dus enkel voor nieuwe data.

Gedachtengang / Hoe zijn we tot de oplossingen gekomen

We hebben de oorspronkelijke manier om de databank te vullen gebruikt, dus aan de hand van een ORM. We hebben de data ingelezen en vervolgens de data toegevoegd aan de databank. We gebruiken telkens een python script, hier wordt de cleanup (uit de vorige epic) reeds gedaan. Dit zorgt ervoor dat de data uit het CSV bestand en de data uit de databank dezelfde structuur hebben.

Epic 3

Korte uitleg van de epic

Op basis van de data van een gegeven contactID wordt er een lijst van toekomstige campagnes voorgesteld. Deze voorspelling wordt gemaakt door een machine learning model dat getraind is op historische data van gelijkaardige contacten en campagnes.

Beperkingen en uitdagingen

De uitdagingen lagen vooral bij het voorbereiden van de dataset waarop het model zou getraind worden. Er moesten keuzes gemaakt worden hoe de data geaggregeerd zou worden en welke kolommen er gebruikt of niet gebruikt zouden worden.

Bepaalde keuzes door beperkingen

Doordat onze focus lag op het zo veel mogelijk aggregeren van de data, hebben we een aantal kolommen niet gebruikt. Zo hebben we bijvoorbeeld de kolommen met de details van een afspraak niet gebruikt, maar enkel de thema's van de afspraken. Ook hebben we de kolommen met de details van een sessie niet gebruikt, maar enkel de thema's van de sessies. Dit hebben we gedaan omdat we de data wilden aggregeren per contact en per campagne, en niet per afspraak of per sessie.

Gedachtengang / Hoe zijn we tot de oplossingen zijn gekomen

Het plan bij de preprocessing was om drie geagregeerde datasets te creëren, een met alle data per contact, een met alle data per campagne en een met alle informatie over de marketing naar een contact over een campagne (de cdi tables). Omdat het doel van de epic is om campagnes te voorspellen voor een contact, is de contact dataset geagreggeerd per uniek contactID, dus alle data van een contact staat in één rij. Hetzelfde

werd gedaan voor de campagne dataset maar dan op campagneID. De afspraken werden per contactID opgeteld en gegroepeerd per thema, door deze manier zijn de details per afspraak weggevallen. Omdat een campagne meerdere sessies had, en de thema's per sessie soms verschillend waren, werd hierop dezelfde methode toegepast als bij de afspraken. Bij de derde dataset werd er niks speciaals gedaan, deze was een gejoinde versie van de cdi tabellen. De volgende stap was een cross join tussen de contact en campagne dataset, zo kregen we een dataset met alle mogelijke combinaties van contactID's en campagneID's. In de tabel inschrijvingen stonden alle inschrijvingen van een contact voor een campagne, met deze informatie konden we de target kolom maken. De target kolom is een boolean die aangeeft of een contact zich heeft ingeschreven voor de gekoppelde campagne. De derde tabel, met cdi informatie, werd gejoined met de cross join tabel en er werd een 'marketing pressure' uitgerekend. Dit is de totale interactie een contact had met de gekoppelde campagne in de cross join tabel. De uiteindelijke dataframe die werd gebruikt om het model te trainen was de cross join tabel met de target kolom en de marketing pressure kolom. Deze dataframe had nu genoeg voorbeelden van contacten die zich wel en niet hebben ingeschreven voor bepaalde campagnes. Met 50% van de rijen target 0 en 50% target 1, was de dataset gebalanceerd om een model te trainen.

We hebben verschillende modellen getest, zowel supervised als unsupervised. De supervised modellen die we hebben getest zijn:

- LinearSVC
- SGDClassifier
- Logistic Regression
- Decision Tree Classifier
- Random Forest Classifier
- Voting Classifier (hard en soft voting)
- Bagging Classifier
- AdaBoost Classifier
- Gradient Boosting Classifier
- Stacking Classifier (met verschillende final estimators)

Voor al deze modellen werden de hyperparameters getuned met behulp van GridSearchCV.

We hebben één unsupervised model getest, K-means. Deze hebben we niet verder uitgewerkt en getuned omdat de supervised modellen betere resultaten gaven.

Bij het selecteren van het model hebben we gekeken naar accuracy en precision op de test set, die 20% van de data bevatte. Het beste model was een Random Forest Classifier met een accuracy van 86% en een precision van 85%.

Bij de effectieve voorspelling in onze Streamlit applicatie wordt heel de preprocessing herhaald voor de meegegeven contacten omdat er data wordt gebruikt uit ons online datawarehouse op het vic. Voor elk meegegeven contact wordt een voorspelling gemaakt voor elke campagne in het datawarehouse. Alle voorspellingen worden gesorteerd op zekerheid en de top 10 wordt getoond aan de gebruiker.

`Welke data / parameters zijn er gebruikt`

- plaats
- subregio
- ondernemingstype
- ondernemingsaard

- activiteitsnaam
- afspraak thema
- campagne type
- campagne soort
- sessie thema
- cdi visit first visit
- cdi visit total pages
- cdi mail sent clicks
- cdi mail sent
- Inschrijvings tabel (target)

Waarvan is er te weinig data

Door de cross join hebben we veel data gegenereerd, maar er mocht meer informatie zijn over campagnes en sessies ervan.

Epic 4

Korte uitleg van de epic

Het doel van epic 4 was dat een keyuser voor een contact met weinig transacties een lookalike met veel transacties kan identificeren. Ook zou er dan een clustering gemaakt moeten worden van contactpersonen qua jobinhoud, type bedrijf, voorkeuren en (verwacht) gedrag. Hier hebben we dan ervoor gezorgd dat er op basis van een gegeven contactID en een gekozen campagne van dit contact, lookalikes worden gegenereerd. Dit zijn contactpersonen die het meest lijken op het gegeven contact. Ook worden deze gesorteerd van meest naar minst gelijkend op het gegeven contact.

Beperkingen en uitdagingen

Een uitdaging was het mogelijk maken dat er een lookalike gegenereerd kon worden op basis van een campagne. Dit had de klant als feedback gegeven.

Bepaalde keuzes door beperkingen

Er zijn niet superveel campagnes en contacten van Oost-Vlaanderen. Daarom hebben we gekozen om gebruik te maken van cosinus similariteit van de keyphrases van de contactpersonen. Dit is een eenvoudige manier om de contactpersonen te vergelijken met elkaar en zo de meest gelijkende te vinden zonder dat we een model moeten trainen.

Gedachtengang / Hoe zijn we tot de oplossingen gekomen

We hebben besloten om eerst alle data van contact, campagnes, afspraken en inschrijvingen samen te voegen. Daarna hebben we de dataset gevectoriseerd met TF-IDF. Vervolgens hebben we de cosinus similariteit berekend tussen de vector van het gegeven contact en alle andere contacten.

Welke data / parameters zijn er gebruikt

De gebruikte Machine Learning technieken zijn:

- TFIDF-Vectorization
- Cosinus Similariteit

Voor beide technieken hebben we gebruik gemaakt van de Scikit-Learn library.

De gebruikte kolommen zijn:

- DimAccount
  - plaats, subregio, ondernemingstype, ondernemingsaard
- DimContact
  - contactID, functietitel, functieNaam
- DimActiviteit
  - activiteitNaam
- DimCampagne
  - campagneID, campagneType, campagneNaam, campagneSoort
- DimAfspraak
  - keyphrases

Waarvan is er te weinig data

Er is niet zozeer te weinig data, maar er kan gekozen worden om meer kolommen te gebruiken zodat de lookalikes nog beter overeenkomen met het gegeven contact.

Epic 5

Korte uitleg van de epic

Het doel van Epic 5 was dat een keyuser voor een campagne een lijst met contacten kan genereren volgens de waarschijnlijkheid om in te schrijven voor die campagne. Ook moest ervoor gezorgd worden dat bij de sortering contacten met weinig marketing pressure bevoordeeld worden ten opzichte van contacten met een hoge marketing pressure.

Wij hebben ervoor gekozen om op basis van een gegeven campagneID een aantal contactpersonen aan te bevelen die het meest geschikt zouden zijn voor deze bepaalde campagne. Deze contactpersonen worden dan gesorteerd van lage naar hoge marketing pressure.

Beperkingen en uitdagingen

Er was niet veel data beschikbaar om een goed model mee te trainen. Ook zijn er soms meerdere contactpersonen per bedrijf, wat het een extra uitdaging maakte om de data voor te bereiden voor het aanbevelingssysteem.

Bepaalde keuzes door beperkingen

Vanwege het beperkte beschikbare data hebben we besloten om gebruik te maken van de cosinus similariteit van de keyphrases van de contactpersonen. Dit is een eenvoudige manier om de contactpersonen met elkaar te vergelijken en zo de meest geschikte te vinden zonder de noodzaak om een model te trainen.

Gedachtengang / Hoe zijn we tot de oplossingen zijn gekomen

Na het uitproberen van verschillende clustering-modellen, waaronder het K-Nearest Neighbours (KNN) algoritme en K-Means clustering, alsook de Surprise library, hebben we uiteindelijk gekozen voor TF-IDF-Vectorization en Cosinus Similariteit. Clustering-modellen vertoonden geen duidelijk onderscheid tussen de clusters, en de Surprise library was niet geschikt vanwege de vereiste ratings die niet beschikbaar waren.

In eerste instantie probeerden we keyphrases te embedden met behulp van de Embedding-API van OpenAI, maar de resulterende dimensie van 1536 maakte de datapreprocessing traag en overtrof de capaciteit van onze beschikbare hardware. Vervolgens probeerden we het Tensorflow Universal Sentence Encoder, maar ook dit gaf dimensies van 512, wat nog steeds te groot was en bovendien moeilijker te implementeren dan de OpenAI API. Uiteindelijk hebben we geopteerd voor de TF-IDF Vectorizer van Scikit-Learn vanwege zijn snelle prestaties.

Het evalueren van de effectiviteit van verschillende embeddings was beperkt omdat we geen uitgebreide tests konden uitvoeren op de beschikbare data, en Voka slechts een beperkt aantal tests kon uitvoeren. Voordat we de keyphrases in het aanbevelingssysteem gebruikten, pasten we enkele technieken toe om stopwoorden te verwijderen en de keyphrases te normaliseren, waarvoor we de NLTK library gebruikten.

`Welke data / parameters zijn er gebruikt`

De gebruikte Machine Learning technieken zijn:

- TFIDF-Vectorization
- Cosinus Similariteit

Voor beide technieken hebben we gebruik gemaakt van de Scikit-Learn library.

Om de keyphrases te maken hebben we volgende kolommen gebruikt:

- DimAccount
  - plaats, subregio, ondernemingstype, ondernemingsaard
- DimActiviteit
  - activiteitNaam
- DimFunctie
  - functietitel
- DimAfspraak
  - thema, onderwerp, keyphrases, afspraak_betreft
- DimCampagne
  - naam, type, soort
- DimMailing
  - naam, onderwerp
- DimSessie
  - thema_naam

Deze keyphrases hebben we dan gecleaned door middel van de NLTK library. We verwijderden onder andere de stopwoorden, haalden herhalingen weg, zette alles in lowercase en maakten gebruik van de stemmer.

Om de marketing pressure te bereken hebben we volgende kolommen gebruikt: (In de user interface kan je zelf kiezen welke kolommen je wilt gebruiken)

- Persoon
  - alle kolommen met mail_thema en mail_type, marketingcommunicatie
- CDI_Visit
  - bron, visit_first_page, visit_total_pages
- Mailing
  - bij mailing hebben we de mail_click_frequency berekend per account

Waarvan is er te weinig data

Er is te weinig data beschikbaar voor campagnes en accounts in Oost-Vlaanderen.

Epic 7

Korte uitleg van de epic

In epic 7 is het de bedoeling dat de keyuser een hypergepersonaliseerde mail kan genereren voor een bepaald contact. In deze mail worden er de meest relevante campagnes en diensten weergeven voor dat contact.

De keyuser kan zelf de mail nog aanpassen eens deze gegenereerd is, zelf een handtekening toevoegen die vervolgens zal worden opgeslagen in de cache en het aantal voorgestelde campagnes bepalen.

Beperkingen en uitdagingen

De grootste uitdaging voor deze epic was het gebruik van een reeds bestaand AI-model zoals ChatGPT. Ondanks de mooie zinnen dat ChatGPT kan genereren was het vrij moeilijk om deze consistent en toepasbaar te krijgen. Deze zinnen zijn vaak te lang, to complex of bevatten teveel onnodige informatie.

Er is gexperimenteerd met andere AI-modellen maar geen enkele voldeed aan onze voorwaarden.

Bepaalde keuzes door beperkingen

Ondanks de tegenslag van deze modellen zijn we met het creatieve idee gekomen om een AI model na te bootsen zonder deze zelf te echt te moeten programmeren. Dit hebben we gedaan aan de hand van libraries die de zinnen die we willen gebruiken bevatten.

Gedachtengang / Hoe zijn we tot de oplossingen zijn gekomen

Het idee voor deze oplossing was dus om een voorbeeld tekst te schrijven ingedeeld in enkele onderdelen. Deze onderdelen zijn belangrijk omdat we op deze manier controle hebben over de structuur en opmaak van onze email. De onderdelen van de email zijn: de begroeting, reden voor contact, de voorgestelde campagnes, de afsluiter en de handtekening.

Voor de onderdelen: begroeting, reden voor contact en de afsluiter, wordt een random zin gekozen uit drie verschillende libraries (afhankelijk van het onderdeel). In deze library zitten zinnen die door ons gecontroleerd en gekozen zijn. We hebben deze zinnen echter niet allemaal zelf geschreven. we zijn te werk gegaan met een voorbeeldzin en aan de hand van deze zin hebben we een aantal andere zinnen laten genereren die de lengte en essentie behouden. vervolgens wordt er per gegenereerde email een random zin gekozen uit elk van deze libraries. Dit zorgt er voor dat niet elke email het zelfde is maar wel telkens dezelfde boodschap heeft.

De voorgestelde campagnes is afhankelijk van de ontvanger van de mail. De keyuser geeft een contact in en kiest het aantal campagnes het in de mail wilt hebben. Aan de hand van het model uit epic 3 worden de aantal campagnes genereerd en worden deze netjes getoont in de mail.

Als laatste maar niet onbelangrijk komt de handtekening. Deze is simpelweg zelf in te geven door de keyuser. Dit hoeft echter maar 1 keer te gebeuren aangezien deze input wordt opgeslagen in de cache en deze handtekening dus zal onthouden.

Welke data / parameters zijn er gebruikt

Buiten het contactId wordt er in deze epic niet rechtstreeks data gebruikt. Wel wordt er gebruikt gemaakt van epic 3 die ,zoals eerder uitgelegd werd, wel wat date gebruikt.

`Waarvan is er te weinig data` Aangezien deze epic geen data rechtstreeks gebruikt is er geen tekort aan data waar genomen.

Epic 8

`Korte uitleg van de epic`

Het doel van epic 8 was om de mogelijkheid te creeren waarbij een key user een lijst van accounts (dus ingeschreven bedrijven) kan genereren die het meest waarschijnlijk zijn om hun lidmaatschap het komende jaar NIET te vernieuwen.

`Beperkingen en uitdagingen`

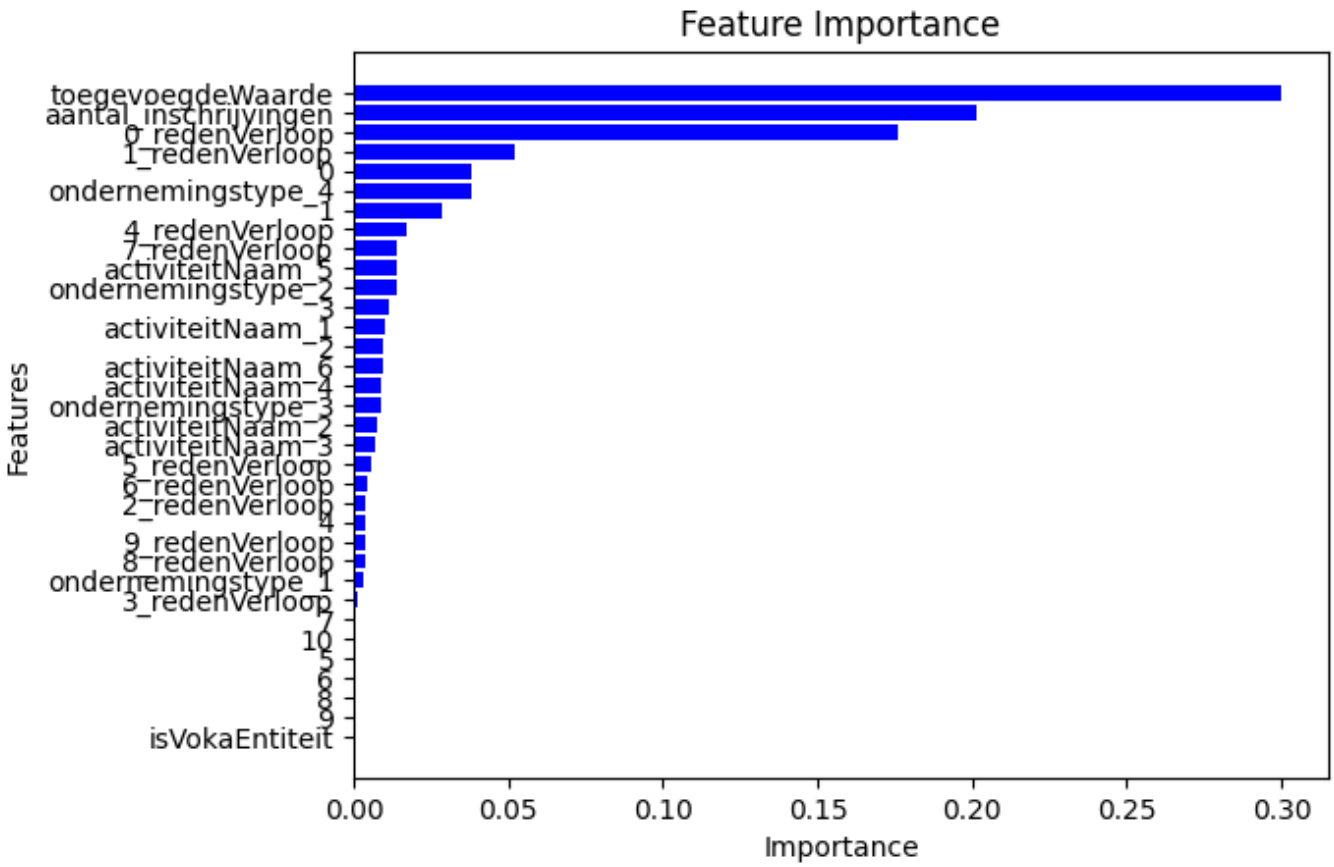
Een redelijke uitdaging bij deze epic was het feit dat er niet veel data was om een model mee te trainen. Dit had als effect dat de accuracy van het model niet heel hoog lag. Een uiteindelijk doel zou zijn om meer data te hebben voor accounts uit de regio Oost-Vlaanderen. Nu hadden we een train_set van iets meer dans 5000 accounts, en een test set van iets meer dan 1000 accounts.

`Bepaalde keuzes door beperkingen`

Ondanks de beperkte hoeveelheid data voor de accounts hebben we toch gekozen voor het trainen van een aantal welgekende Machine Learning modellen. Het resultaat was dus (zoals eerder vermeld) dat er minder accuracy was bij deze modellen (dit hebben we dan gewoon aanvaard). Een andere mogelijke oplossing was geweest om het model te trainen met de data van alle accounts, en niet enkel die van Oost-Vlaanderen. Hierdoor gingen we al een stukje meer data hebben en dus ook een hogere accuracy. Echter was de Data Warehouse waar we mee werken al opgebouwd voor enkel de regio Oost-Vlaanderen, dus hebben we hier gewoon mee verder gewerkt.

`Gedachtengang / Hoe zijn we tot de oplossingen zijn gekomen`

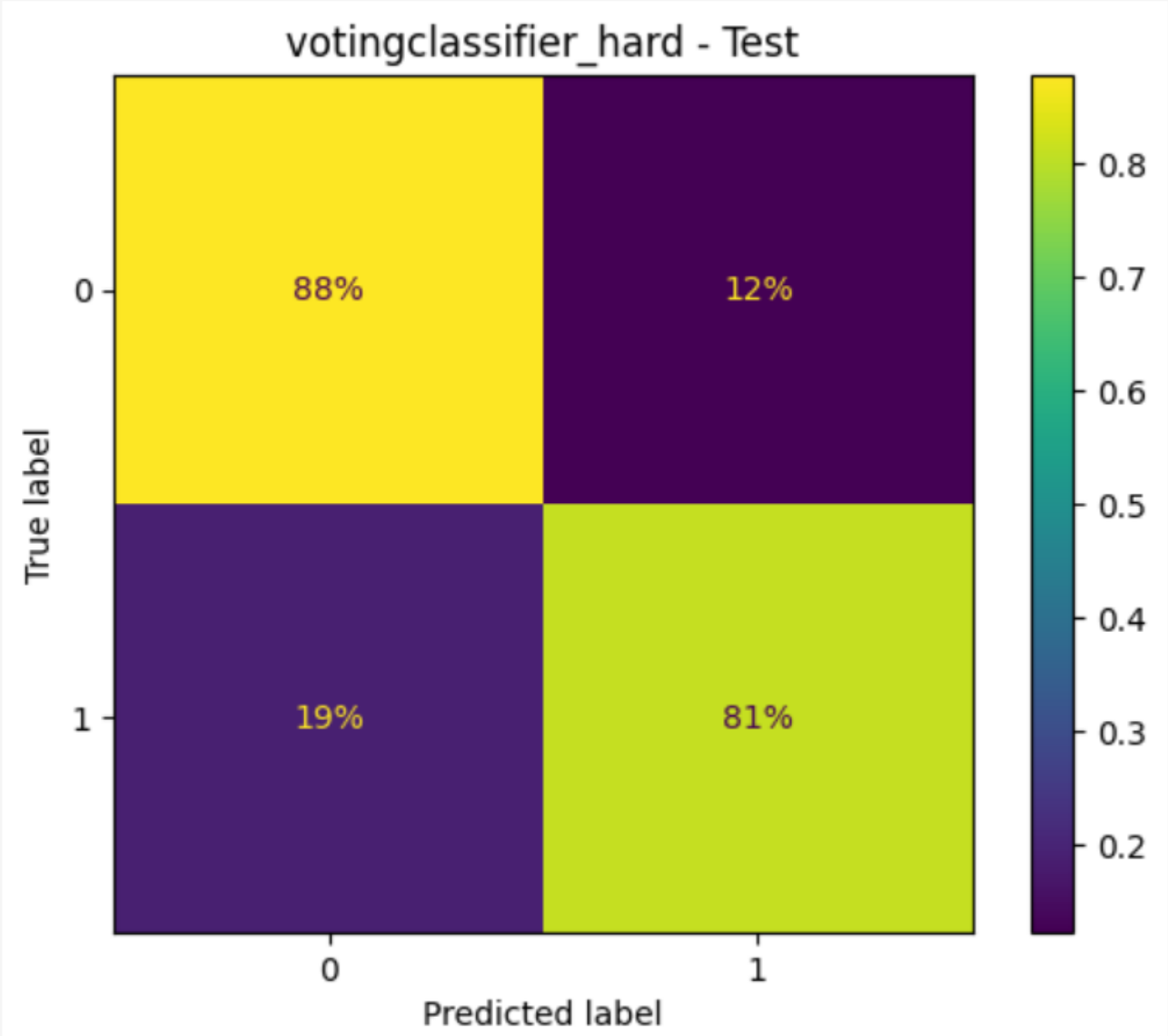
Om tot een model te komen die aan het einddoel voldoet, hebben we eerst de juiste data ingeladen. Dit bedroeg een dataframe van de data van de tabel DimAccount en de data van de tabel DimLidmaatschap. Na analyse van deze gegevens konden we concluderen dat de hoofdreden waarom mensen hun lidmaatschap opzeggen is dat ze 'geen gebruik' maken van hun lidmaatschap. Concreet wil dat dus zeggen dat ze geen/heel weinig inschrijvingen hebben voor campagnes van Voka. Daarnaast leek het ons zo dat de financiële gegevens van de accounts ook een impact zouden moeten hebben op een eventuele stopzetting van het lidmaatschap. Omwille van die reden hebben we ook enkele kolommen van de tabel DimFinanciëleDataAccount toegevoegd aan de dataframe. Vervolgens zijn er een aantal kolommen toegevoegd/gecreerd geweest: boekjaar, aantal_inschrijvingen, lidmaatschap_actief. De lidmaatschap_actief kolom is gebaseerd op de opzegDatum kolom van DimLidmaatschap. Als deze gelijk is aan '2026-1-1' dan zal deze waarde gelijk zijn aan 1, dit wijst er dus op dat voor dit account het lidmaatschap nog actief is (anders 0). Deze kolom is dus ook hetgene dat we willen voorspellen met ons model. De kolom boekjaar bedraagt het jaar VOOR het jaar van de opzegdatum, en de kolom aantal_inschrijvingen bedraagt het aantal inschrijvingen in campagnes per account voor dat boekjaar. Dit is dus een gegeven die ons zicht geeft op het gebruik van het lidmaatschap. Na datacleaning (one hot encoding) hebben we gekozen om enkel te werken/trainen met features die een importance hebben van 0.01 of meer.



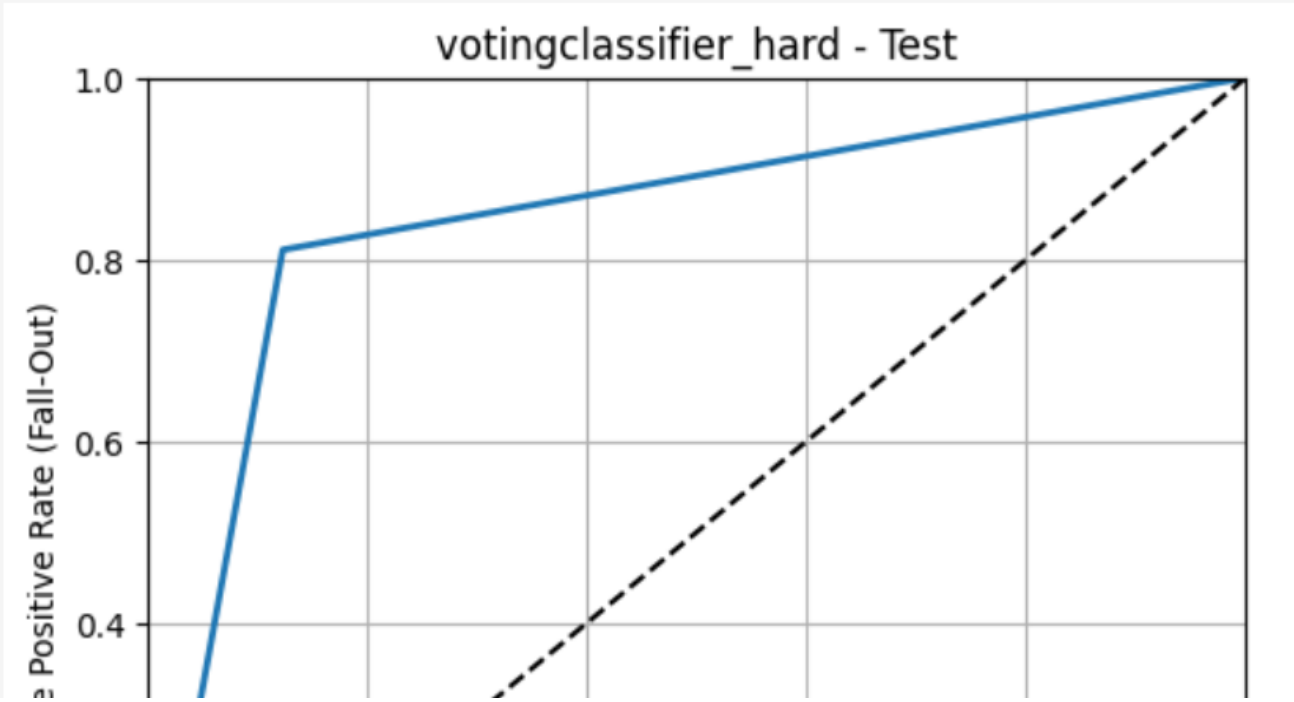
Daarna hebben we een tiental modellen getraind en geëvalueerd.

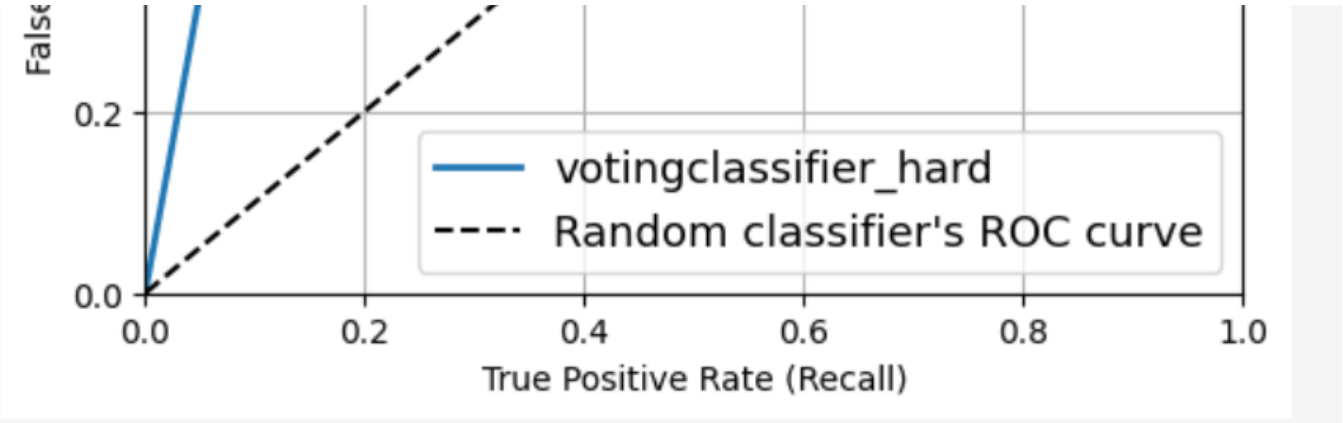
	Name	accuracy	precision	recall	f1	roc_auc
0	linearsvc	0.673684	0.659218	0.240326	0.352239	0.583810
1	sgdclassifier	0.669925	0.541534	0.690428	0.606983	0.674177
2	logisticregression	0.651880	0.523569	0.633401	0.573272	0.648047
3	deciciontreeclassifier	0.830075	0.799097	0.720978	0.758030	0.807449
4	randomforestclassifier	0.854887	0.805328	0.800407	0.802860	0.843589
5	votingclassifier_hard	0.852632	0.794411	0.810591	0.802419	0.843913
6	votingclassifier_soft	0.851880	0.800000	0.798371	0.799185	0.840782
7	BaggingClassifier	0.855639	0.805726	0.802444	0.804082	0.844607
8	AdaBoostClassifier	0.842857	0.798729	0.767821	0.782970	0.827295
9	GradientBoostingClassifier	0.849624	0.797546	0.794297	0.795918	0.838150
10	StackingClassifier_sgd	0.854887	0.804082	0.802444	0.803262	0.844011
11	StackingClassifier_lr	0.855639	0.806982	0.800407	0.803681	0.844185
12	StackingClassifier	0.806015	0.737271	0.737271	0.737271	0.791758
13	StackingClassifier_rf	0.836842	0.771825	0.792261	0.781910	0.827596

Na evaluatie hebben we het model gekozen met de hoogste recall-score. Dit wil zeggen dat het model het minst aantal false negatives heeft. Dit is belangrijk omdat we willen dat het model zo weinig mogelijk accounts mist die hun lidmaatschap zullen opzeggen. Beter een account te veel voorspellen die zijn lidmaatschap niet zal opzeggen, dan een account te missen die zijn lidmaatschap wel zal opzeggen. Het model die bij deze redenering uit de bus kwam was de hard voting classifier met een recall-score van 81%.




```
<Figure size 640x480 with 0 Axes>
```





Welke data / parameters zijn er gebruikt

De geteste Machine Learning modellen zijn:

- LinearSVC
- SGDClassifier
- Logistic Regression
- Decision Tree Classifier
- Random Forest Classifier
- Voting Classifier (hard en soft voting)
- Bagging Classifier
- AdaBoost Classifier
- Gradient Boosting Classifier
- Stacking Classifier (met verschillende final estimators)

De uiteindelijke gebruikte Machine Learning modellen zijn (voor gebruikt model):

- One Hot Encoding
- Grid Search
- Hard Voting Classifier
- Confusion Matrix
- ROC Curve

Voor deze technieken hebben we gebruik gemaakt van de Scikit-Learn library.

De gebruikte kolommen zijn:

- DimAccount
  - accountID, plaats, isVokaEntiteit, ondernemingstype, activiteitNaam
- DimLidmaatschap
  - redenAangroei, redenVerloop, startDatum, opzegDatum,
- DimFinanciëleDataAccount
  - toegevoegdeWaarde (matchend op boekjaar kolom die gecreëerd is)
- Overige
  - boekjaar, aantal_inschrijvingen, lidmaatschap_actief

Waarvan is er te weinig data

Zoals eerder vermeld, is er te weinig data voor accounts uit de regio Oost-Vlaanderen. Dit heeft als gevolg dat de accuracy van het model niet heel hoog ligt. Een uiteindelijk doel zou zijn om meer data te hebben voor accounts uit de regio Oost-Vlaanderen. Of om op algemene schaal te werken en alle regio's in de datawarehouse te gebruiken.

Algemene reflectie

Aangeleverde data en Datakwaliteit

Tijdens dit project zijn er enkele uitdagingen ontstaan door de aangeleverde data. De data vertoonde inconsistenties en bevatte talrijke fouten, waaronder NaN-waarden en verschillende formaten. Om structuur aan te brengen, moesten we regelmatig de kolomnamen van de aangeleverde data aanpassen. Bovendien was er sprake van inconsistente scheidingstekens in de CSV-bestanden, waarbij zowel ';' als ',' werden gebruikt in dezelfde bestanden. Handmatige aanpassingen waren noodzakelijk.

Ook deden zich problemen voor met Foreign Keys en Primary Keys, zoals situaties waarin Foreign Keys naar niet-bestaande Primary Keys verwezen. Daarnaast klopten de datatypes niet altijd, zoals datums die in string-formaat waren in plaats van in DateTime-formaat. Dit proces vergde aanzienlijke tijd. Bij het overzetten van de data naar het Data Warehouse deden zich extra complicaties voor, waardoor het duidelijk werd dat de algehele datakwaliteit niet optimaal was. Het aantal NaN-waarden in de data was aanzienlijk, en na het uitvoeren van data-cleaning bleef er voor sommige epics slechts beperkte bruikbare data over.

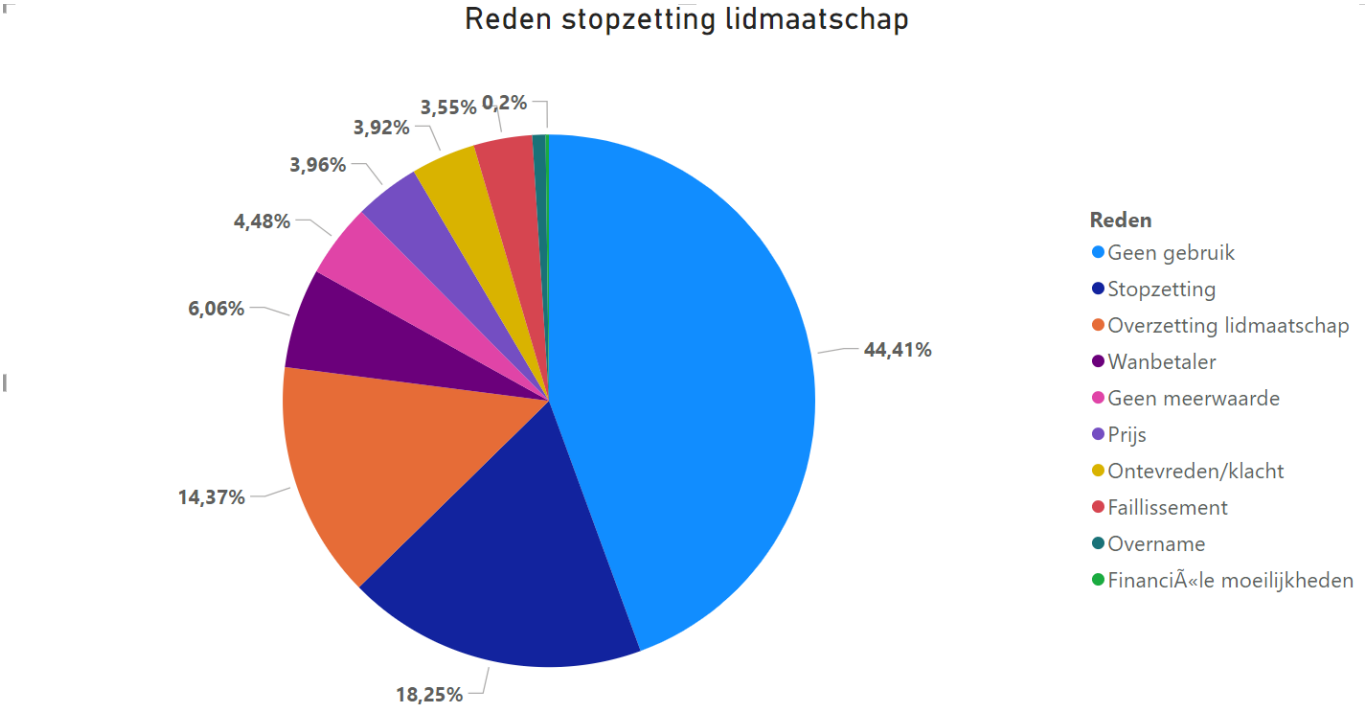
Mogelijkheden / beperkingen om inzichten te verkrijgen

Aan de ene kant hebben we inzichten en resultaten verkregen door analyses in zowel PowerBI als bij het trainen van modellen. Aan de andere kant is het belangrijk op te merken dat wij als studenten niet over de juiste achtergrond beschikken om de data volledig correct te interpreteren. Ondanks onze contactmomenten met de klant, blijkt het voor ons onmogelijk om de diepgaande kennis toe te passen die een interne specialist zou hebben. Bovendien is er een algemene trend in dit project waarbij voor bepaalde epics onvoldoende data overblijft om de gewenste modellen te trainen. Dit gebrek aan data kan leiden tot een vertekend of onjuist resultaat.

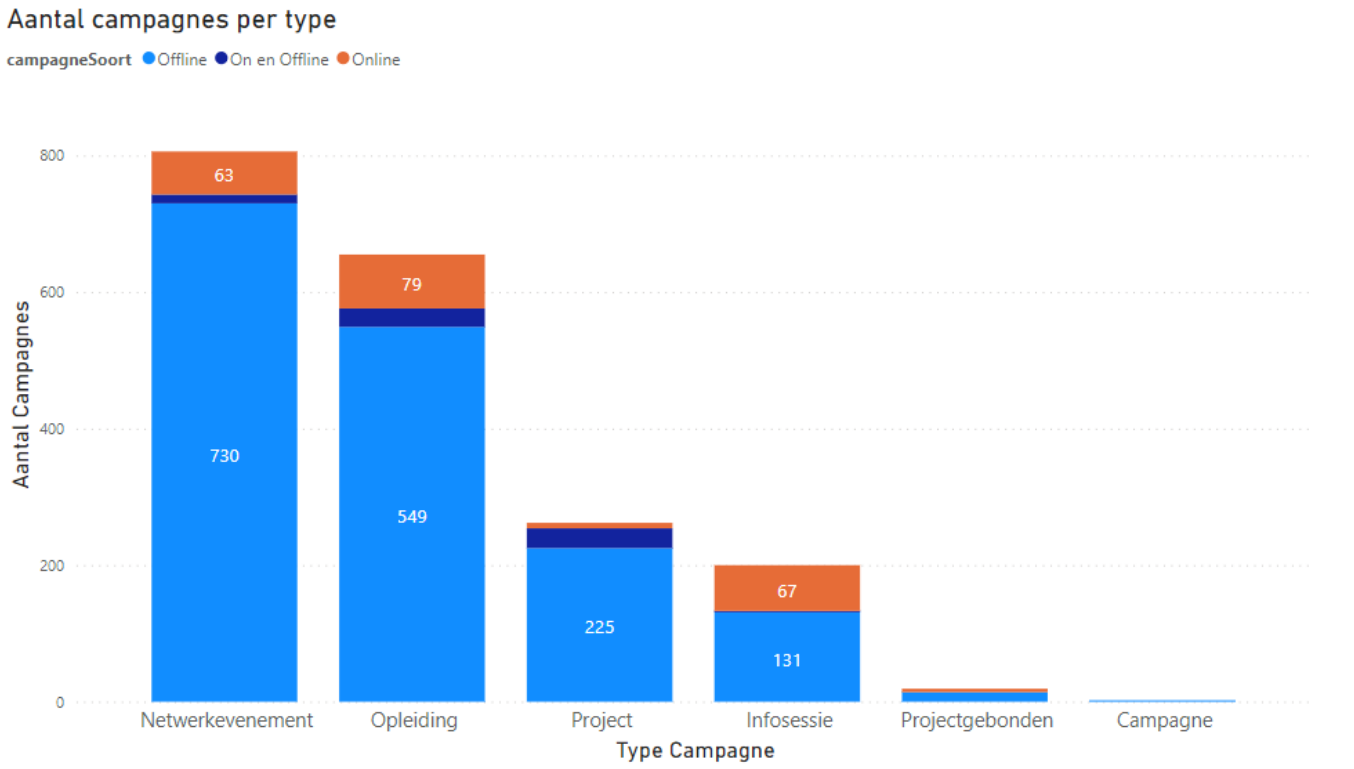
De testfase kon ook niet door ons worden uitgevoerd, aangezien de data geanonimiseerd is. Wel heeft de klant enkele tests uitgevoerd, zij het beperkt. Bijvoorbeeld heeft Voka epic 5 gebruikt om campagnes aan te bevelen voor 20 van hun klanten. Eén klant heeft vervolgens op deze mail gereageerd en zich ingeschreven voor de aanbevolen campagne. Dit resultaat is echter niet optimaal voor een aanbevelingssysteem. Mogelijke redenen hiervoor kunnen zijn: het model is niet zo optimaal als gehoopt, de klanten hebben de mail niet gelezen, of de klanten worden afgeschrikt door het gebruik van AI bij de aanbevelingen. Hieruit kunnen we concluderen dat kwalitatieve testen van essentieel belang zijn. Zonder deze testen kunnen we niet met zekerheid zeggen dat de verkregen resultaten ook daadwerkelijk correct zijn.

Verkregen inzichten in PowerBI

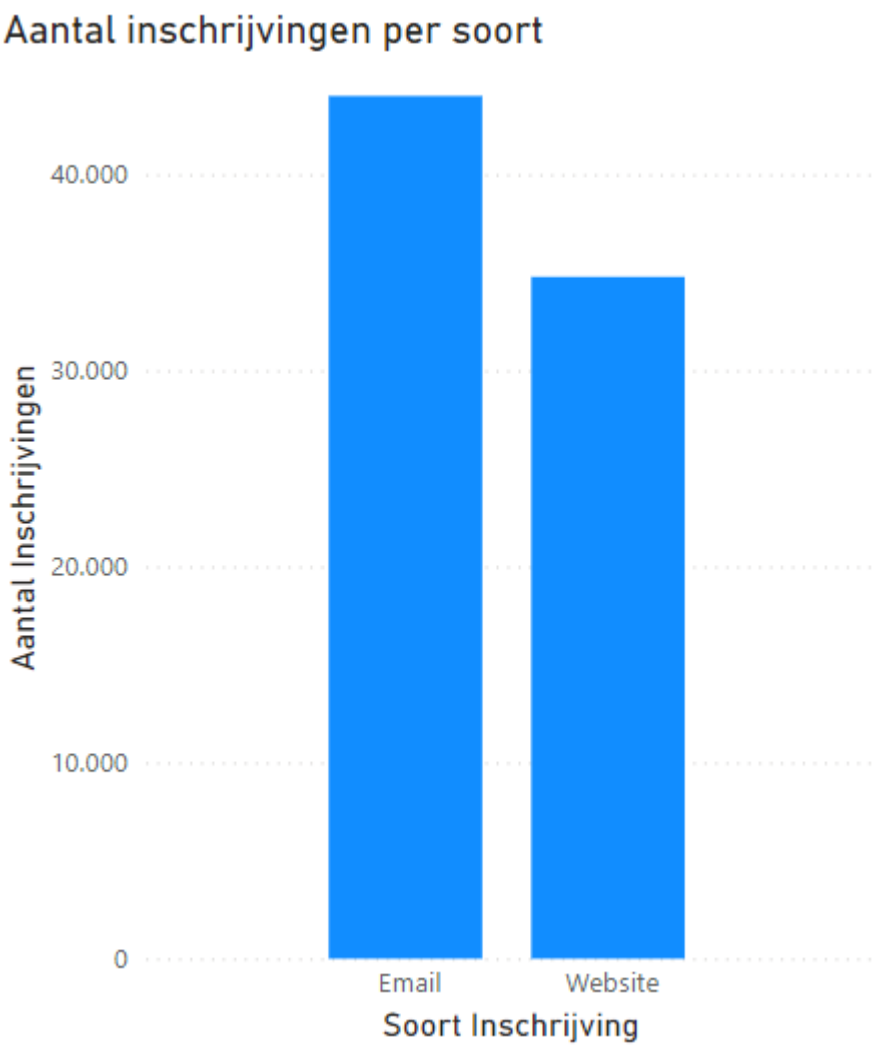
Uit onze PowerBI-analyse zijn toch redelijk veel inzichten naar voren gekomen. Hier delen we er enkele die ons het meest zijn opgevallen, en die we het meest interessant vonden.



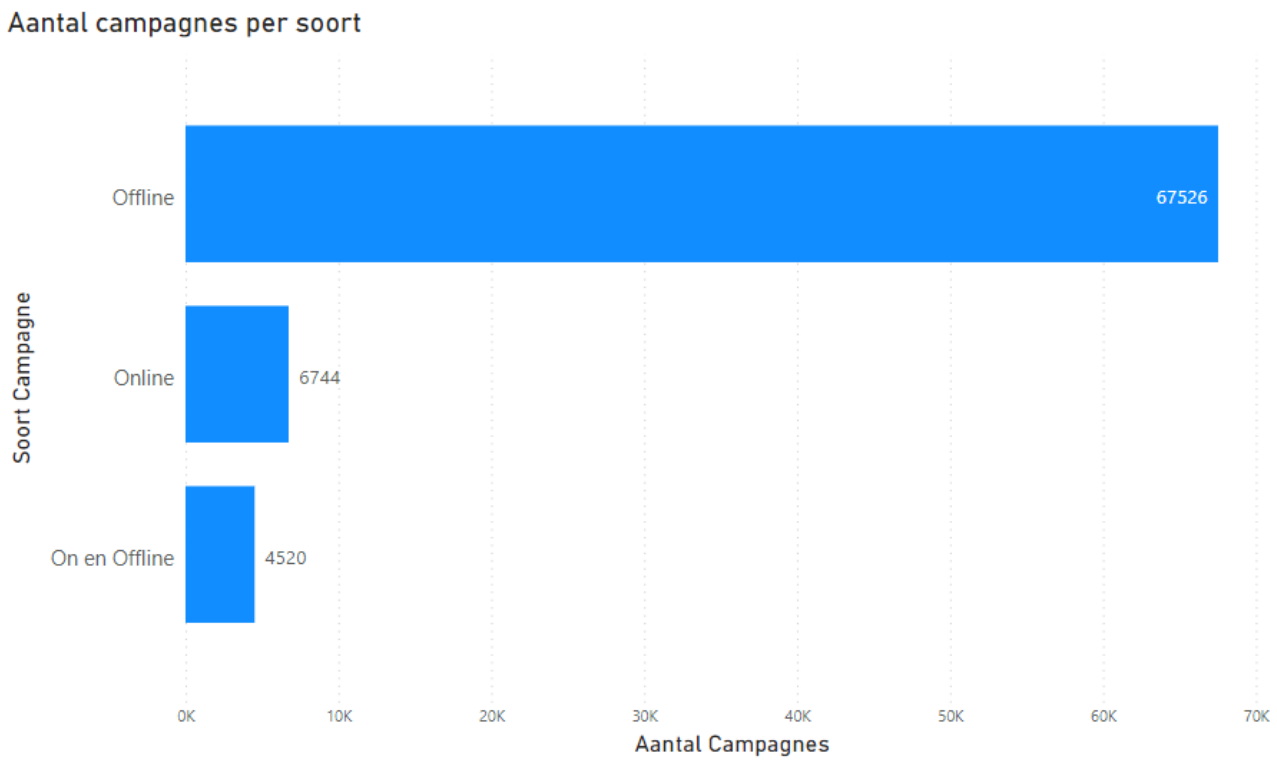
Op deze grafiek zien we de verschillende redenen waarom klanten van Voka in het verleden beslist hebben de samenwerking/het lidmaatschap stop te zetten.



Op deze grafiek zien we de verschillende type van campagnes die Voka aanbiedt. Ook is hier een onderscheid gemaakt in online en offline campagnes.



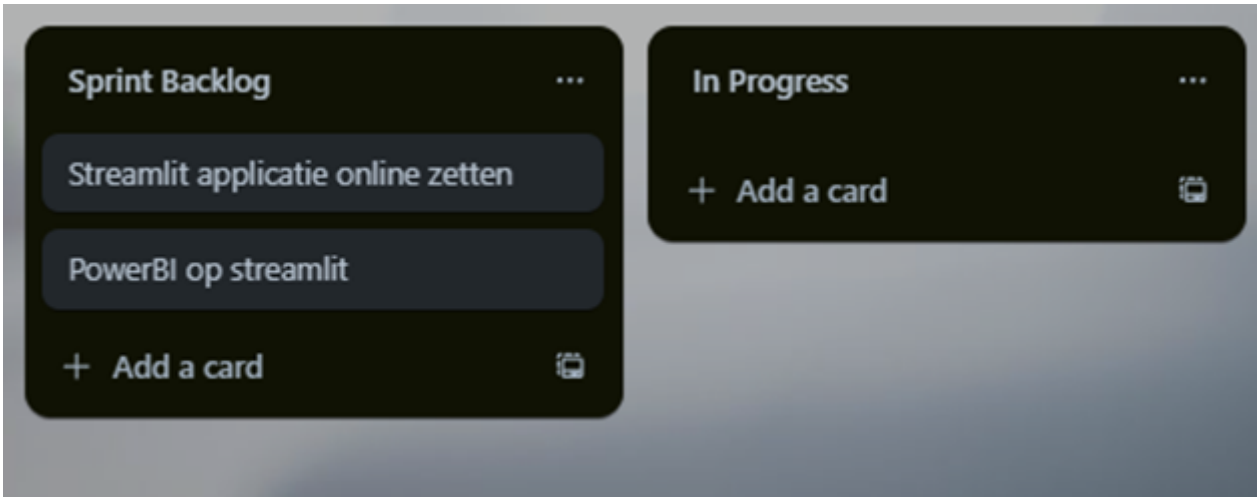
Deze grafiek toont het totaal aantal inschrijvingen in campagnes opgesplitst per soort: via e-mail of via website.



Deze grafiek toont de hoeveelheid campagnes per soort.

Sprintrapport sprint 6

`Product Backlog, wat ingepland was en werd gedaan`



`Wat hebben we meer gedaan dan gepland en waarom?`

Wij hadden gepland om deze sprint te gebruiken voor algemene cleanup en afwerking. Ook hebben we deze sprint ons einddossier volledig uitgewerkt.

`Wat hebben we minder gedaan dan gepland en waarom? Hoe kunnen we dit beperken om op schema te blijven?`

Niets, we hadden gepland om deze sprint als ‘reserve’ te zien zodat we nog eventuele problemen konden oplossen en code konden organiseren. We hebben de Streamlit applicatie niet online beschikbaar kunnen maken omdat er op het VIC geen poort voorzien is. Het is dus niet mogelijk dat je via het VIC een poort

openstelt op het internet. Ook hebben we onze PowerBI rapporten niet in onze Streamlit applicatie kunnen integreren omdat je hier bepaalde toelatingen nodig hebt van je Microsoft account beheerder.

Team Progressie

Aangezien alle voorziene epics waren afgewerkt zoals gehoopt, konden we deze sprint gebruiken als ‘extra’ om de nodige cleanup en afwerking uit te voeren. Ook hebben we samen het einddossier geschreven waar alle details over de uitgewerkte epics in terug te vinden zijn. Over het algemeen was ook tijdens deze sprint de groepssfeer en samenwerking zeker optimaal. We hebben geen nieuwe problemen ondervonden hierbij en de communicatie over zowel de afwerking als het einddossier verliep vlot.