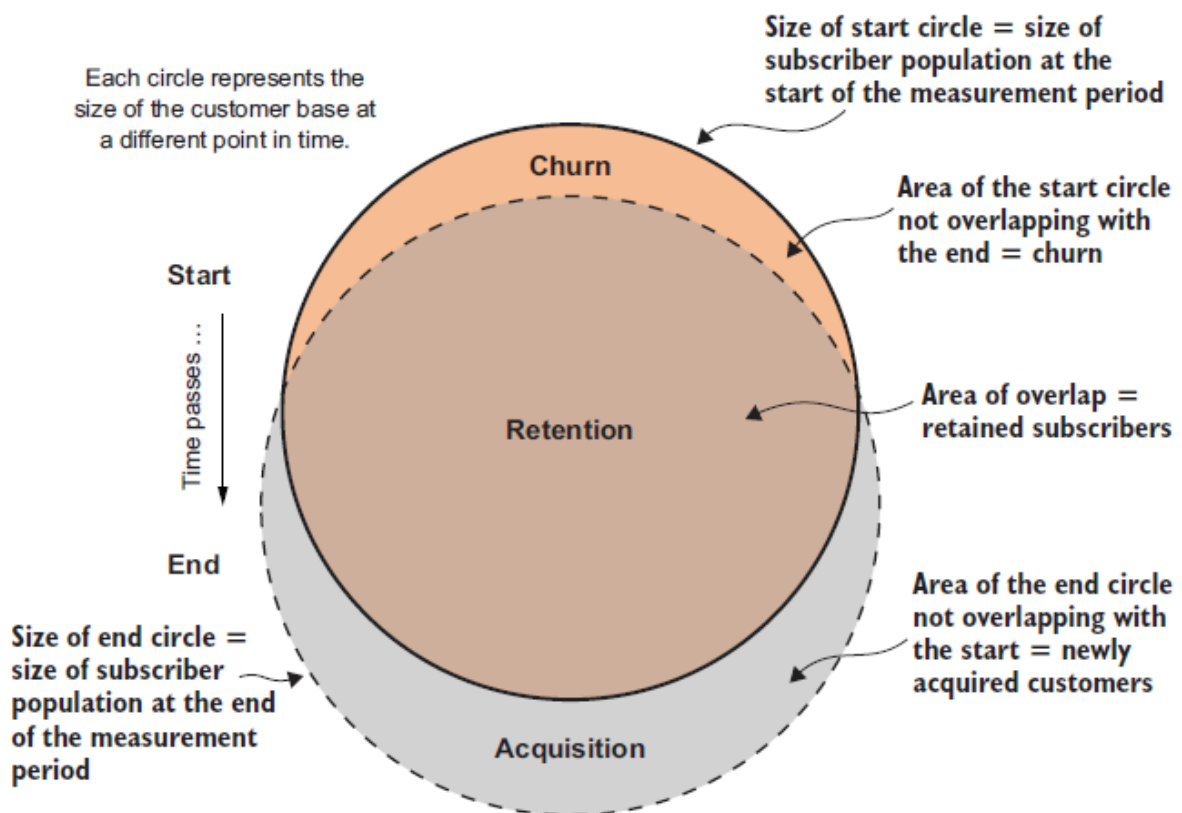


Pós-Graduação em Data Science & Business Analytics  
UC Data Science Project

Turma I02 Grupo 13  
JAIR OLIVEIRA  
JOAO FERNANDES

### Descrição Projeto

O objetivo do projeto é analisar o Dataset de uma empresa com clientes com compras recorrentes ou por subscrição. Dessa análise devemos encontrar quais as variáveis que permitem que os clientes mantenham as suas compras recorrentes ou subscrições, e dessa forma criar um modelo preditivo capaz de prever quais clientes irão abandonar este processo, e deixar de comprar os produtos ofertados por esta organização. Este tipo de análise permite que a organização defina estratégias a fim de ampliar a base de clientes e reduzir a perda dos que possui. Esquemáticamente podemos definir o nosso problema com o modelo seguinte.



## 1- Relatório Fase 1

Este documento visa apresentar o Relational Model e o ETL identificado no nosso Projeto Final. O nosso projeto tem como base um Dataset de Vendas de uma empresa farmacêutica, os dados foram disponibilizados no Kaggle. Este Dataset é constituído por uma tabela de vendas, com dados de abril de 2019 até setembro de 2019, a nível de item de fatura com mais de 1 milhão de *samples*. O objetivo do projeto será criar um modelo de *Machine Learning* que permita prever quais os clientes que vão repetir as compras para o mês de outubro de 2019. Para isso, há também uma tabela com os clientes que repetiram as compras no mês de outubro de forma a podermos avaliar o modelo.

Também é objetivo desta fase do Projeto criar um Data Warehouse para armazenar os dados do Dataset de forma a podermos criar diversos relatórios sobre o negócio desta empresa.

A estrutura desta fase do relatório intermédio consiste na apresentação do *Relational Model* a qual contém a *Bus Matrix* com as *Dimensions Tables* e as *Facts Tables* que constituem o Data Warehouse. Na segunda parte será apresentado o processo de ETL com as transformações efetuadas no Azure Data Factory contendo a ingestão dos dados, transformações e por último o store dos dados no Data Warehouse apresentado no Dimensional Model.

### 1.1- Dimensional Model

#### - Bus Matrix

A *Bus Matrix* consiste num esquema em forma de matriz, onde estão identificados os principais processos do negócio, as suas granularidades e métricas. Ao avaliarmos os requisitos do negócio, identificámos 2 processos diferentes:

1. Vendas – Com dados de vendas de de Abril a Setembro.
2. Vendas Outubro – Com dados relativos a informação de quais clientes compraram em Outubro.

As granularidades e métricas foram definidas para suprir as análises ao menor nível e responder às questões identificadas em cada processo. Também é definido na *Bus Matrix* todas as *Dimensions* e as suas relações com cada processo, como apresentado na tabela seguinte.

Tabela 1. Bus matrix.

Processo de Negócio	Granularidade	Métricas	Data	Loja	Cliente	Medic	Pagamen
Vendas	1 linha por linha de fatura	Quantidade e Valor total	X	X	X	X	X
Vendas_Outubro	1 linha por Cliente	1 e 0 para compra e não compra	X		X		

### - Dimension Tables

As *Dimension Tables* caracterizam-se por conter o contexto textual relacionado a cada medição do processo de negócio, ou seja, os atributos e características que servirão de restrições de consulta, agrupamentos e rótulos.

Abaixo apresentamos todas as *Dimension Tables* (**Tabela 2**) identificadas na *Bus Matrix*, com todos os atributos necessários para cada dimensão conforme a necessidade para o negócio.

**Tabela 2.** Dimension tables.

Dim_Data	Dim_Medico	Dim_Cliente	Dim_Loja	Dim_pagamento	
DataID	Medico Key	Cliente Key	Loja Key	PagamentoID	
Data	MedicoID	ClienteID	LojaID	Descricao	
Descricao da data	Nome	Nome	Nome	Data Efetiva da Linha	
Dia da Semana	NIF	NIF	Endereco	Data de Expiracao da Linha	
Mes	Endereco	Endereco	Cidade	Indicador de Linha Atual	
Ano	Cidade	Cidade	Regiao		
Feriado Indincador	Regiao	Regiao	Codigo Postal		
	Codigo Postal	Codigo Postal	Numero Telefone		
	Numero Telefone	Numero Telefone	Data Efetiva da Linha		
	E-mail	Email	Data de Expiracao da Linha		
	Data Efetiva da Linha	Data Efetiva da Linha	Indicador de Linha Atual		
	Data de Expiracao da Linha	Data de Expiracao da Linha			
	Indicador de Linha Atual	Indicador de Linha Atual			

### - Fact Tables

Neste ponto, apresentamos todas as Tabelas Factuais relacionadas a cada processo, respeitando as granularidades e métricas definidas.

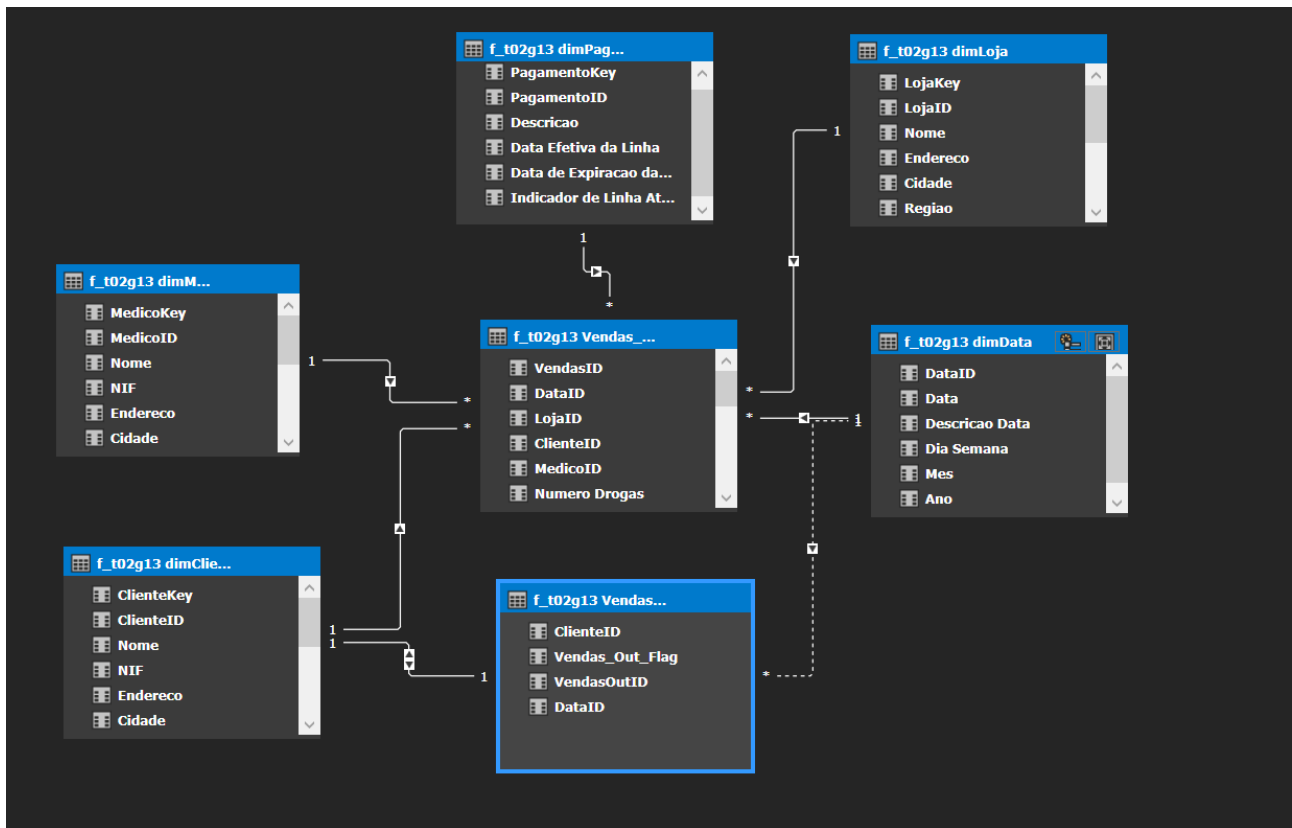
**Tabela 3.** Fact tables.

Vendas_Fact		Vendas_outubro_Fact
VendasID		VendasOutubroID
DataID		ClienteID
LojaID		DataID
ClienteID		Oct_purchase_flag
MedicoID		
PagamentoID		
Numero Drogas		
QTD Total		
Valor Total		
QTD Receita		
QTD Generica		
QTD Cirurgica		
QTD Alternativa		
QTD Geral		
QTD OTC		
QTD Cronica		
QTD Aguda		
QTD H1		

## - Star Schema

Neste ponto apresentamos o modelo estrela do nosso Dimensional Model à qual apresenta as ligações entre as *Dimensional Tables* e *Fact Tables*.

Fig 1 – Star Schema

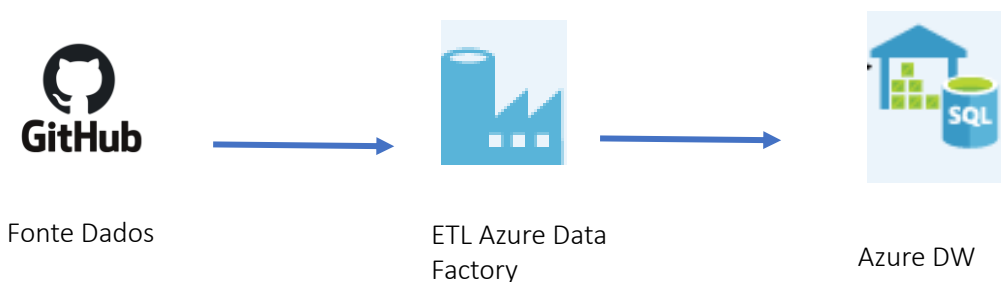


## 1.2- ETL

O ETL é o processo desenvolvido pelas ferramentas de Data Integration das organizações. Tipicamente consiste nas seguintes três etapas: extração, transformação, carregamento. Desta forma vamos obter uma visão consolidada dos dados, de forma a poder tomar as melhores decisões de negócio.

O objetivo é integrar dados de múltiplos sistemas e fontes e consolidá-los num único ambiente. Neste caso, armazenaremos no Data Warehouse cuja estrutura foi apresentada no ponto anterior.

Nesta etapa, utilizou-se o Azure Data Factory para orquestrar todo o processo de ETL. Utilizou-se o GitHub como fonte dados RAWs, Azure Data Lake para armazenar os dados e processar as respetivas transformações. Posteriormente, os dados, foram disponibilizados no Azure SQL Database.



## - Desafios

O Dataset original apresentava algumas insuficiências, tendo em conta as necessidades deste Projeto Final. Seguidamente vamos elencar quais as dificuldades encontradas e a forma de as contornar.

### I. Falta de Dimensões

O Dataset tinha um conjunto de dados em que muitas das dimensões identificadas apenas tinham um ID. Tornou-se necessário enriquecer os dados, simulando a existência de atributos que possam dar profundidade às dimensões. Assim, a partir de dados originais foi usado código Python para enriquecer esses dados, gerando atributos a partir da concatenação de IDs com outros atributos. Os dados finais foram armazenados no GitHub.

As imagens abaixo contém exemplos de criação de duas dimensões (Data e Médico).

Imagem 2- Make\_dim\_date

```
def make_dim_date(df):

    df["created_at_bill"] = pd.to_datetime(df["created_at_bill"])
    df["Date"] = df["created_at_bill"].map(lambda x: x.strftime('%Y-%m-%d'))
    df["Date"] = pd.to_datetime(df["Date"])
    df["DateID"] = df["Date"].map(lambda x: x.strftime('%Y%m%d'))

    # Criando as features da tabela dim_date
    dim_date = df[["DateID", "Date"]]
    dim_date["Full Date Description"] = df["Date"].map(lambda x: x.ctime())
    dim_date["Day Of Week"] = df["Date"].map(lambda x: x.strftime("%A"))
    dim_date["Month"] = df["Date"].map(lambda x: x.strftime("%B"))
    dim_date["Year"] = df["Date"].map(lambda x: x.strftime("%Y"))

    dim_date = dim_date.drop_duplicates(subset=["DateID"], keep='first')

    df.drop(["created_at_bill"], axis=1, inplace=True)

    return dim_date
```

Imagem 3- Make\_dim\_doctor

```
def make_dim_doctor(df):

    df["Doctor Name"] = df["doctor_ref_id"].map(lambda x: "Doctor " + str(x))

    dim_doctor = df_train[["doctor_ref_id", "Doctor Name"]]
    dim_doctor["Doctor Key"] = df["doctor_ref_id"]
    dim_doctor["NIF"] = df["doctor_ref_id"].map(lambda x: "00" + str(x))
    dim_doctor["Address"] = df["doctor_ref_id"].map(lambda x: "Rua " + str(x))
    dim_doctor["City"] = df["doctor_ref_id"].map(lambda x: "Cidade " + str(x))
    dim_doctor["Region"] = df["doctor_ref_id"].map(lambda x: "Region " + str(x))
    dim_doctor["Zip Code"] = df["customer_ref_id"]
    dim_doctor["Phone Number"] = df["customer_ref_id"].map(lambda x: "+351 00" + str(x))
    dim_doctor["Address E-mail"] = df["customer_ref_id"].map(lambda x: "email" + str(x) + "@gmail.com")
    dim_doctor["Row Effective Date"] = df["Date"]
    dim_doctor["Row Expiration Date"] = "2999-12-31"
    dim_doctor["Current Row Indicator"] = 1

    dim_doctor = dim_doctor.drop_duplicates(subset=["doctor_ref_id"], keep='first')

    df.drop(["Doctor Name", "Date"], axis=1, inplace=True)

    return dim_doctor
```

O código completo encontra-se no Notebook “create\_dimensions.ipynb”.

## II. Repositório para Armazenamento

Outro ponto de grande dificuldade foi encontrar um repositório que fosse possível conectar ao Azure Data Factory através de um endereço HTTP.

Inicialmente os dados foram armazenados no Kaggle, porém não conseguimos realizar o *scraper* do *link* à qual os arquivos foram armazenados. Portanto, precisávamos armazenar os dados no GitHub, uma vez que o mesmo fornece o *link* de *Raw data* de maneira simples.

No entanto, outro problema surgiu, o Github armazena arquivos no máximo com 25 MB. Como algumas das tabelas eram superiores a 100 MB houve necessidade de efetuar o *split* dos dados. Para isso também se utilizou *scripts* em Python para realizar o *split*.

Imagem 4 – Split dados em Azure ML

```
#Split em 5 datasets
df_train1, df_train2, df_train3, df_train4, df_train5 = \
    np.split(df_train, 5)

df_train1.shape, df_train2.shape, df_train3.shape, df_train4.shape, df_train5.shape
((236805, 21), (236805, 21), (236805, 21), (236805, 21), (236805, 21))
```

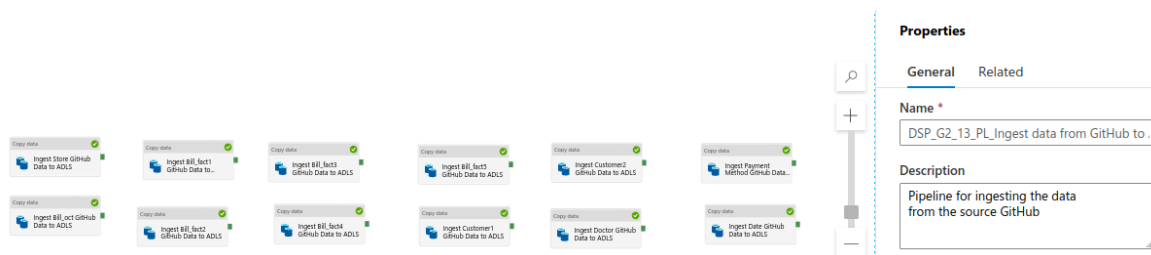
### ETL - Azure Data Factory

O Azure Data Factory é um orquestrador para o processo de ETL ou ELT, possui integração com mais de 90 fontes de dados e tem uma fácil utilização. Foram criadas algumas pipelines para realização de todo o processo.

#### - Extract

A extração dos dados foi realizada através da Pipeline “DSP\_G2\_13\_Ingest data from GitHub to ADLS”. Esta Pipeline contém as “Copy” das tabelas para Azure Data Lake, conforme imagem 5.

Imagem 5 – Pipeline Ingest data from GitHub to ADLS



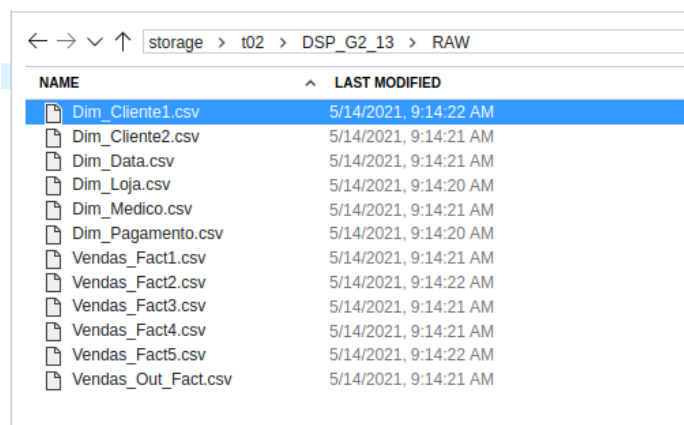
A fonte de dados foi o GitHub Conectado com o Linked service através de um HTTP .

Imagem 6 – Linked service fonte de dados



Os dados foram armazenados no ADLS conforme é possível observar na imagem 7.

Imagem 7 – Dados armazenados no RAW ADLS



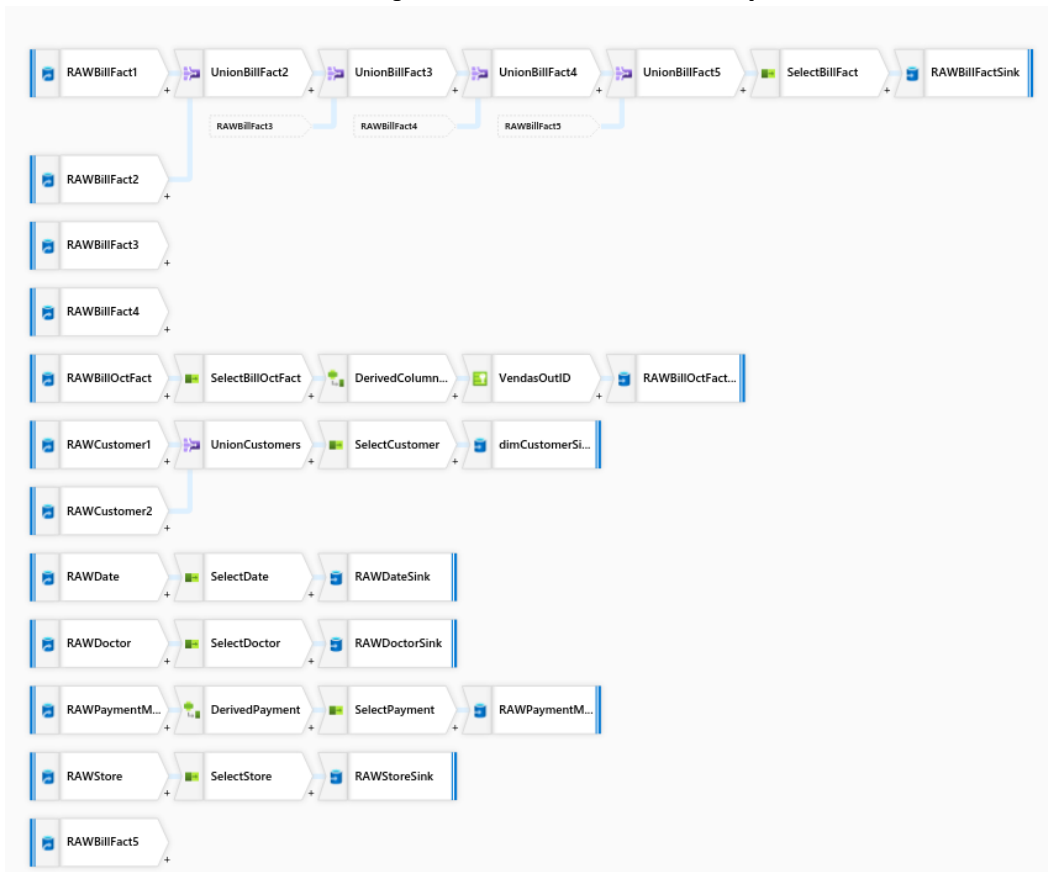
NAME	LAST MODIFIED
Dim_Cliente1.csv	5/14/2021, 9:14:22 AM
Dim_Cliente2.csv	5/14/2021, 9:14:21 AM
Dim_Data.csv	5/14/2021, 9:14:21 AM
Dim_Loja.csv	5/14/2021, 9:14:20 AM
Dim_Medico.csv	5/14/2021, 9:14:21 AM
Dim_Pagamento.csv	5/14/2021, 9:14:20 AM
Vendas_Fact1.csv	5/14/2021, 9:14:21 AM
Vendas_Fact2.csv	5/14/2021, 9:14:22 AM
Vendas_Fact3.csv	5/14/2021, 9:14:21 AM
Vendas_Fact4.csv	5/14/2021, 9:14:21 AM
Vendas_Fact5.csv	5/14/2021, 9:14:22 AM
Vendas_Out_Fact.csv	5/14/2021, 9:14:21 AM

### - Transform

Está etapa foi realizada através do *Data Flow contido na Pipeline “DSP\_G2\_13\_PL\_Ingest data from RAW to PROCESSED”* onde estão contidas todas as transformações necessárias. Os dados neste *Data Flow* são copiados do *Raw*, transformados e gravados numa instância PROCESSED com os seus devidos nomes.

Nesta fase tentou-se antecipar todas as modificações necessárias para as etapas futuras do projeto. Foram realizadas as operações de alteração de tipos de dados, renomear, criação de colunas, exclusão de colunas etc.

Imagem 8 – Data Flow de transformações



Os dados transformados foram salvos em pastas criadas com respectivo nome.

Imagem 9 – Dados Processed

storage > t02 > DSP_G2_13 > PROCESSED	
NAME	LAST MODIFIED
Bill_fact	5/11/2021, 8:19:16 PM
Bill_Oct_Fact	5/13/2021, 7:35:04 PM
dimCustomer	5/13/2021, 7:41:39 PM
dimDate	5/13/2021, 7:42:26 PM
dimDoctor	5/13/2021, 7:42:11 PM
dimPaymentMethod	5/13/2021, 7:42:41 PM
dimStore	5/13/2021, 7:41:51 PM

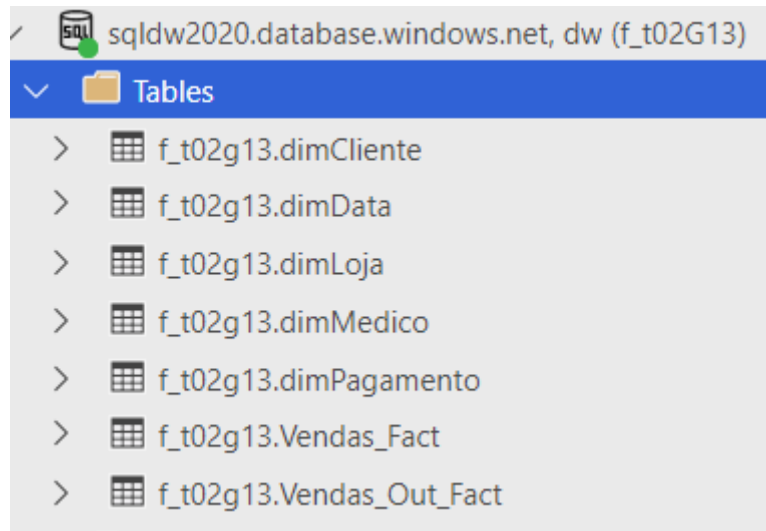
### - Load

Neste último passo foi feito o Egress dos dados para o DW numa Azure SQL Database, para isso foi criado um *pipeline* "DSP\_G2\_13\_PL\_Egress data from ADLS to DW". O DW foi criado utilizando as informações enviadas pelo professor ao grupo 13.

O Resultado do DW criado pode ser observado na imagem em a qual contém a conexão ao Azure Data Studio.

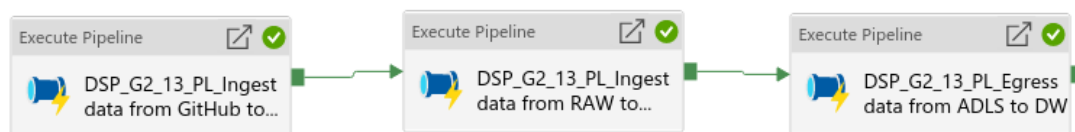


Imagem 10 – DW e tables criadas



No final, todo o processo foi adicionado a uma única Pipeline “DSP\_G2\_13\_PL\_Main Pipeline” que será objeto de entrega desta fase 1.

Imagem 11 – Main Pipeline



## 2- Relatório Fase 2

Nesta segunda fase apresentamos o Modelo Semântico e o Relatório em Power BI em continuação as etapas do projeto.

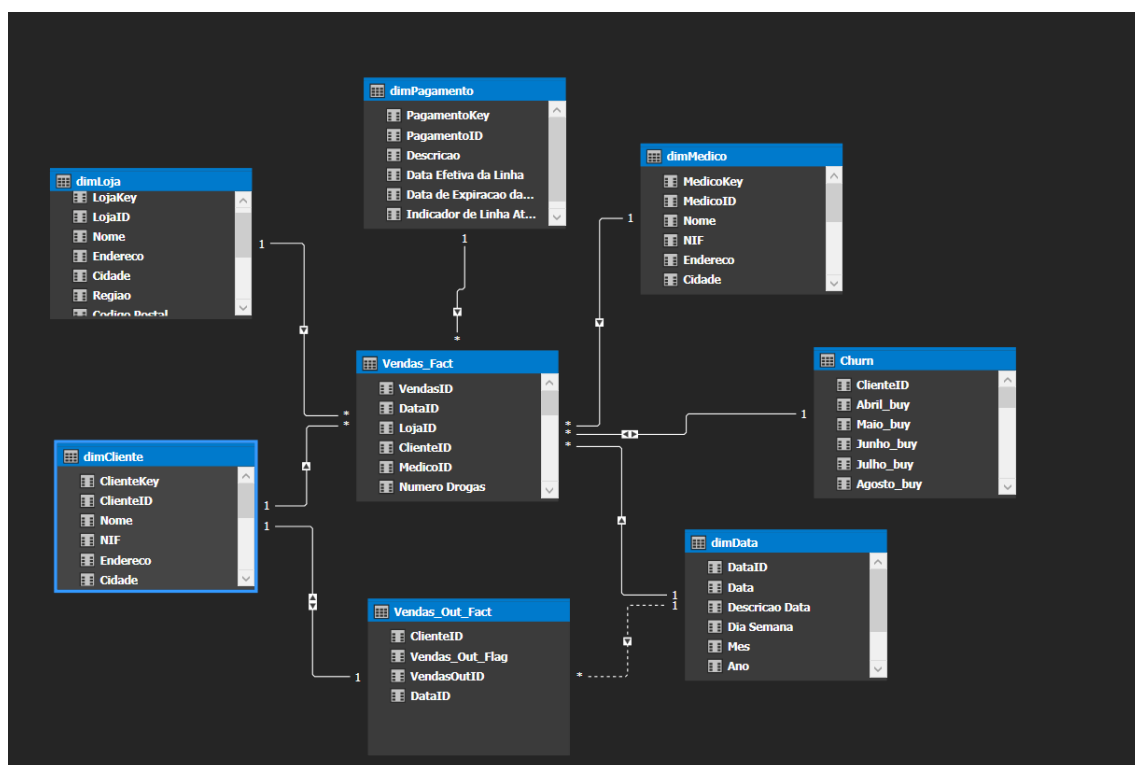
### 2.1- Modelo Semântico

O Modelo Semântico é um instrumento de análise e design de bases de dados. Ele fornece uma metodologia para estruturar os dados e as relações entres os dados. Um objeto semântico define-se como uma colecção de atributos que só por si descreve uma entidade. Estes atributos são os dados referentes ao objecto semântico que se pretende armazenar na base de dados.

Neste Projeto utilizou-se o Analysis Services para criar o modelo semântico que suportará a elaboração dos relatórios no Power BI.

Na figura 11 pode-se ver o esquema do modelo semântico em estrela com as ligações entre todas as dimensões e fact tables. Desta forma, pode-se assim realizar queries entre as diferentes tabelas.

Imagem 11 – Esquema do Modelo Semântico



- Tabelas calculadas- Churn

Nesta etapa precisou-se criar tudo que seria necessário para a análise que será realizada em PowerBI, para isso definiu-se a coluna data e assim seria possível toda análise que envolvesse a componente temporal. Em seguida, criou-se algumas colunas e measures.

Outro ponto importante é perceber informações em relação aos dados de churn, para isso foi criado a tabela de Churn que serve de base ao nosso projeto, daí ter sido desde logo criada no modelo semântico, nesta tabela foram também calculadas algumas measures mais relevantes e irão ser utilizadas no Power Bi e fornecer informações de perda e retenção de clientes ao longo de cada mês.

Imagem 12 – Tabela de Churn

[illegible]

Seguidamente foi feito o Deploy do nosso modelo, tendo sido executado com sucesso.

Imagem 13 – Deploy do modelo

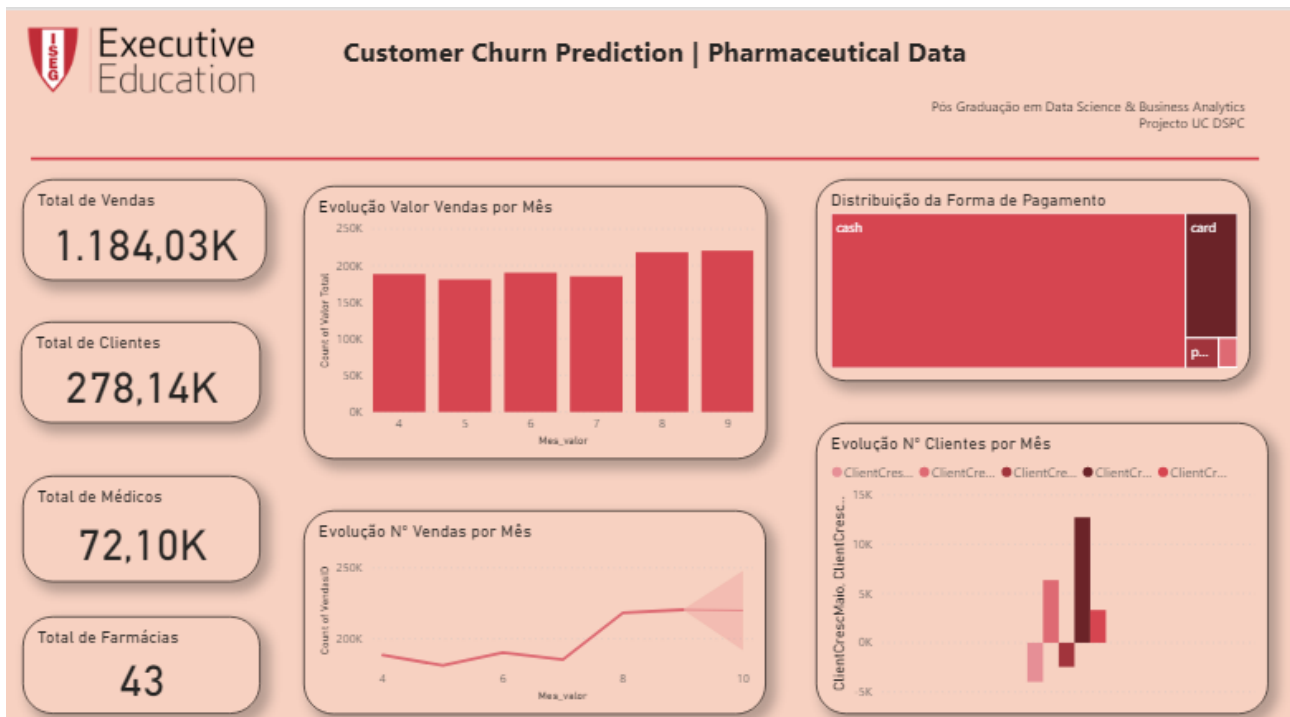
The image shows a screenshot of a data table and a console output window. The table at the top has columns with values like 6243108, 0, 83360, 20191, and 83361. Below the table is a console output window with the following text:

```
Output
Show output from: Build
Build started...
----- Build started: Project: DSP_ModelSemantic_V1, Configuration: Development x86 -----
----- Deploy started: Project: DSP_ModelSemantic_V1, Configuration: Development x86 -----
----- Build: 1 succeeded or up-to-date, 0 failed, 0 skipped -----
----- Deploy: 1 succeeded, 0 failed, 0 skipped -----
```

## 2.2 – Power BI

Seguidamente apresentamos algumas imagens do nosso relatório em Power BI. Dividimos a nossa apresentação em três partes, sendo que a primeira é uma apresentação geral e descrição do dataset que serviu de base a este projeto.

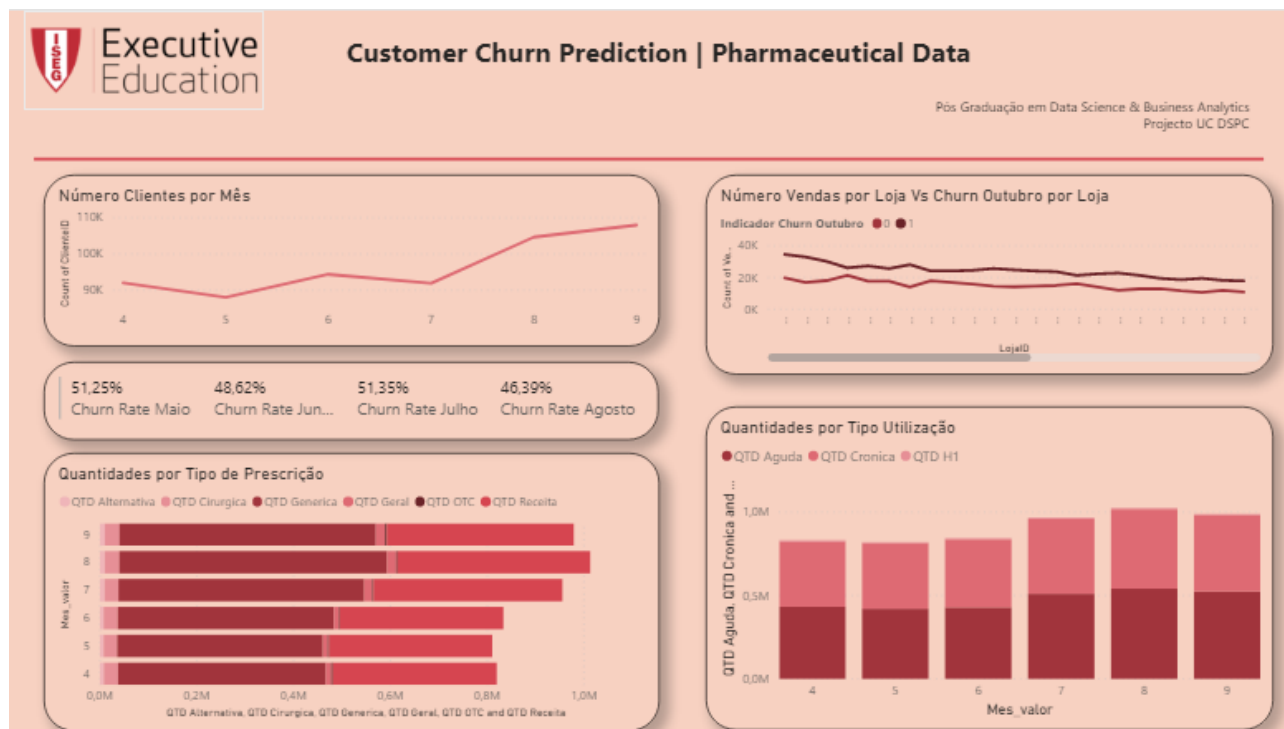
Imagem 14 Relatório Power BI Definição Geral



O nosso Dataset apresenta as vendas de uma empresa farmacêutica numa cadeia de lojas (farmácias), com uma base de clientes bastante alargada, bem como o conjunto dos médicos que prescrevem os produtos desta empresa farmacêutica. Verifica-se que as vendas têm subido nos meses em análise, no entanto também se verifica uma grande alteração do número de clientes que compram mensalmente, pelo que se pretende inferir quais os fatores que poderão determinar que um cliente desista de comprar os produtos, ou que as estas compras sejam feitas de forma irregular. Nesta primeira página também é apresentada uma descrição de qual o meio de pagamento utilizado nas compras, pelos clientes. Verifica-se que a compra a Cash é dominante em grande parte das transações pelo que este facto é afastado como explicativo para o Churn.

Na segunda parte o Dataset é analisado numa perspetiva mais específica das dimensões cliente e vendas.

Imagem 15 –Relatório Power BI – Análise Cliente e Vendas



O número de clientes, apesar de alguma irregularidade, mantém-se crescente, e também se verifica que nas farmácias onde as vendas são maiores o Churn tende a ser menos acentuado. Nesta segunda parte também tentamos mostrar alguns insights que poderão conduzir aos fatores diferenciadores do Churn. Os produtos vendidos por esta empresa podem ser classificados segundo dois grandes critérios. Por tipo de Prescrição, e neste caso a prescrição com Receita e de produtos Genéricos são os elementos com frequências mais elevadas. E por tipo de Utilização, e neste ponto, a utilização dos medicamentos para doenças com Dor Aguda (Doença momentânea, mas intensa) e Doenças Crónicas são elementos mais prescritos. Foram estes os fatores que foram tomados como determinantes para o Churn e são analisados na terceira parte da apresentação.

Imagem 16 Relatório Power BI Análise das features de Quantidade



Pela análise dos gráficos notamos que a distribuição do Churn é bastante uniforme pelos critérios escolhidos. No entanto, parece haver uma tendência para haver Churn nas prescrições de Produtos Genéricos bem como nos produtos destinados a Doenças Crónicas. De notar que a base alargada de clientes torna esta análise, visualmente, difícil. A grande amplitude das quantidades vendidas por cliente leva-nos a crer que neste Dataset existem, não só clientes consumidores finais como também clientes institucionais. No Dataset inicial não existe nenhuma característica que nos permita identificar esta diferença, pelo que a análise efetuada teve em conta a quantidade média de forma a atenuar este efeito. No entanto, havendo diferença da tipologia do cliente pensamos que o Dataset devia ser dividido, pois as causas do Churn são diferentes para as tipologias de clientes diferentes. Tentaremos confirmar estas tendências no Modelo de Machine Learning.

### 3 – Relatório Fase 3

Nesta fase iniciamos a preparação e construção de um modelo de Machine Learning capaz de prever se um cliente irá ou não realizar *churn*. Essa informação é imprescindível para que as decisões a nível de negócio sejam tomadas com o intuito de melhorar a taxa de retenção de clientes e aumentar a taxa de aquisição.

## - Criação do Dataset

Inicialmente possuímos um conjunto de dados com 1.18 milhões de samples as quais trazem informações a nível de linha de fatura de compra de 278 mil clientes ao longo de 6 meses. Também possuímos informações noutra conjunto de dados a qual traz a classificação se determinado cliente fez ou não churn em outubro do mesmo ano, inclusive essa será nossa label.

O principal desafio foi transformar os dados e criar um Dataset que serviria de base para construção do modelo, neste ponto tivemos que lidar com a componente temporal e encontrar uma forma de transformação e assim eliminar a componente temporal que traria imensa dificuldade de análise. Esta decisão foi discutida e aconselhada nas aulas de acompanhamento deste Projeto Final.

A alternativa adotada foi criar um Dataset agrupando Clientes\_Ids por mês e realizando a soma das demais colunas, logo, adicionamos as informações de cada mês em colunas diferentes, também criamos uma coluna a qual fornece informações da quantidade de compras que cada cliente realizou e assim eliminamos a componente temporal, por último fizemos o join com a nossa label que refere se cada cliente realizou ou não compra no mês de outubro, desta forma o shape do nosso Dataset agora possui 278 mil linhas e 82 colunas.

O código com as transformações realizadas pode ser visto no ficheiro “Churn\_Model\_Prediction” do tipo Jupyter Notebook. Neste ponto demos como concluída a transformação do Dataset, para base de construção do modelo.

## Modelo

Numa primeira análise dos dados destacamos que havia desbalanceamento da variável preditora e alguns missing values gerados no agrupamento das colunas por Cliente\_ID, que foram tratados substituindo por 0. Também realizamos a normalização dos dados com a intuição de diminuir a necessidade de computação para construção dos modelos, neste caso realizamos a normalização separada a nível de dataset de treino e teste de forma a evitar bias ou potenciais bias na avaliação do modelo.

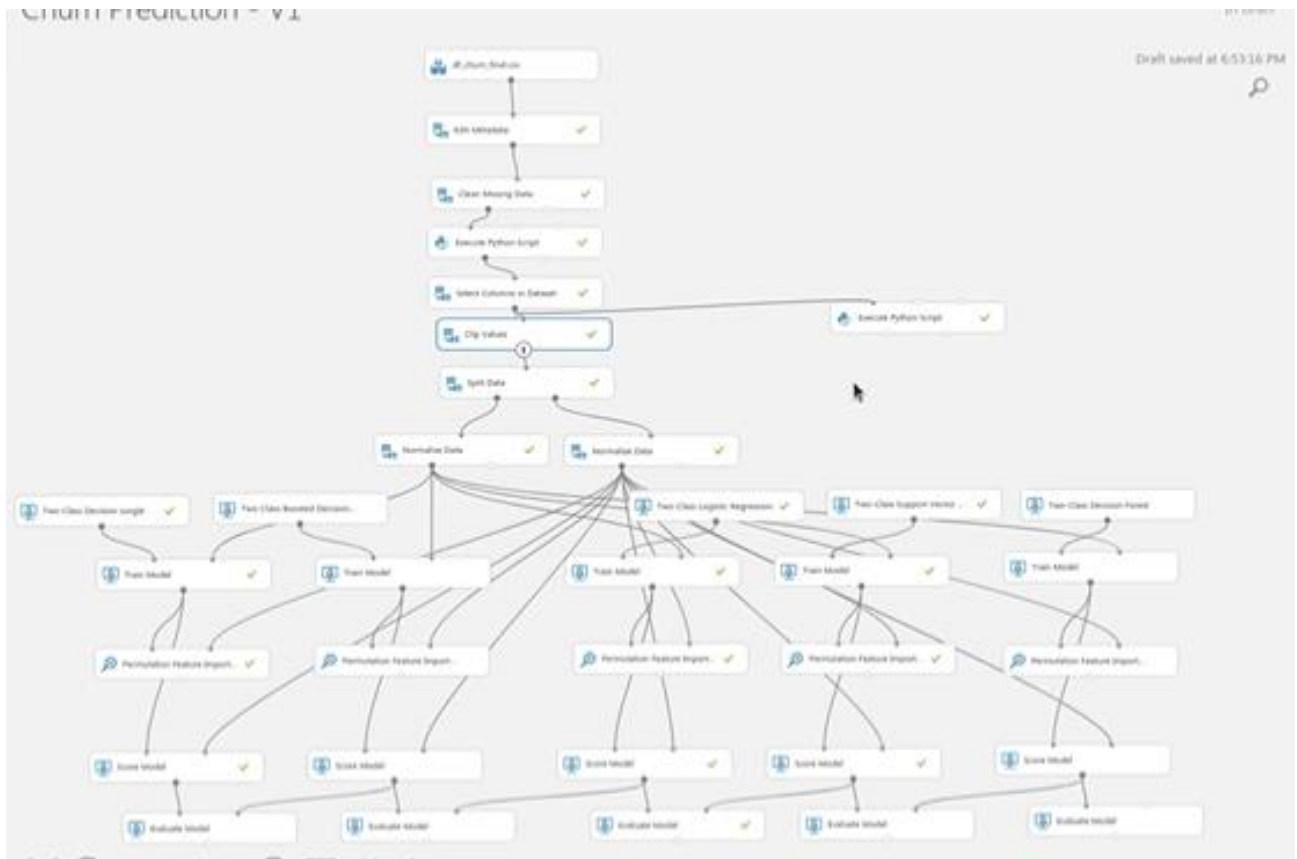
Outro ponto importante foi o tratamento dos possíveis valores Outliers, neste caso identificamos muitos Clientes\_Ids com diversos valores dezenas de vezes maiores que a média. Uma causa provável é que esses clientes podem ser clientes institucionais, embora não tenhamos evidência desta afirmação.

Uma ideia seria tratar esses clientes de forma diferente ou até mesmo num modelo específico caso seja possível. No entanto, decidimos manter esses clientes para não perdemos consistência do Dataset.

Com o nosso *Dataset* base colocámos como primeira hipótese a utilização das 82 features e 5 algoritmos de classificação binária conforme imagem.



Imagem 17 – Azure Machine Learning – Projeto Inicial



Os resultados obtidos com este modelo são os indicados na figura seguinte

Imagem 18 – Azure Machine Learning Projeto inicial Resultados

DataSet Original (82 features)			
	Accuracy	Precision	Recall
Two Class Decision Jungle	0.711	1.000	0.000
Two-Class Boosted Decision Tree	0.679	0.449	0.497
Two-Class Logistic Regression	0.711	1.000	0.000
Two-Class Support Vector Machine	0.694	0.470	0.472
Two-Class Decision Forest	0.712	0.805	0.001

Os resultados obtidos não foram muito satisfatórios pelo que resolvemos tratar o DataSet diminuindo o número de features, indo ao encontro daquilo que foram os insights obtidos no relatório de Power BI.

Sendo assim as features, num\_drugs\_bill, total\_quantity\_bill, total\_spend\_bill, quantity\_surgical, quantity\_ayurvedic, quantity\_general, quantity\_otc, quantity\_h1 e contador foram sumarizadas, retirando a dimensão temporal destas features.

O código Python usado é o que se exemplifica na imagem em baixo.

Imagem 19– Azure Machine Learning Código Python-

```
10 def azureml_main(df):
11
12     df["num_drugs_bill"] = df["num_drugs_bill_4"] + df["num_drugs_bill_5"] + df["num_drugs_bill_6"] + \
13         df["num_drugs_bill_7"] + df["num_drugs_bill_8"] + df["num_drugs_bill_9"]
14
15     df["total_quantity_bill"] = df["total_quantity_bill_4"] + df["total_quantity_bill_5"] + \
16         df["total_quantity_bill_6"] + df["total_quantity_bill_7"] + \
17         df["total_quantity_bill_8"] + df["total_quantity_bill_9"]
18
19     df["total_spend_bill"] = df["total_spend_bill_4"] + df["total_spend_bill_5"] + \
20         df["total_spend_bill_6"] + df["total_spend_bill_7"] + \
21         df["total_spend_bill_8"] + df["total_spend_bill_9"]
22
23     #df["quantity_ethical"] = df["quantity_ethical_4"] + df["quantity_ethical_5"] + \
24     #    df["quantity_ethical_6"] + df["quantity_ethical_7"] + \
25     #    df["quantity_ethical_8"] + df["quantity_ethical_9"]
26
27     #df["quantity_generic"] = df["quantity_generic_4"] + df["quantity_generic_5"] + \
28     #    df["quantity_generic_6"] + df["quantity_generic_7"] + \
```

Após o *Run* deste projeto, os resultados obtidos são mostrados na tabela seguinte

Imagem 20 – Azure Machine Learning Projeto com tratamento de features- Resultados

DataSet Original (35 features)			
	Accuracy	Precision	Recall
Two Class Decision Jungle	0.711	1.000	0.000
Two-Class Boosted Decision Tree	0.635	0.401	0.535
Two-Class Logistic Regression	0.711	1.000	0.000
Two-Class Support Vector Machine	0.718	0.515	0.418
Two-Class Decision Forest	0.711	1.000	0.000

Da tabela de resultados concluímos ainda ter modelos que não generalizavam suficientemente os dados, e por consequência resultavam em indicadores de Precision de 1. Com estes resultados optámos pelo algoritmo de Two- Class Support Vector Machine devido à consistência e equilíbrio dos resultados

### Improvement do Modelo

Para melhorar o modelo usámos inicialmente a operação de Permutation Feature Importance.

Com a tabela de resultados que mostramos em baixo foi decidido retirar a feature contador, já que apresentava um peso de 0 após esta análise.

Imagem 21 – Azure Machine Learning Tabela de Feature Importance

Churn Prediction - 35 features > Permutation Feature Importance > Feature importance

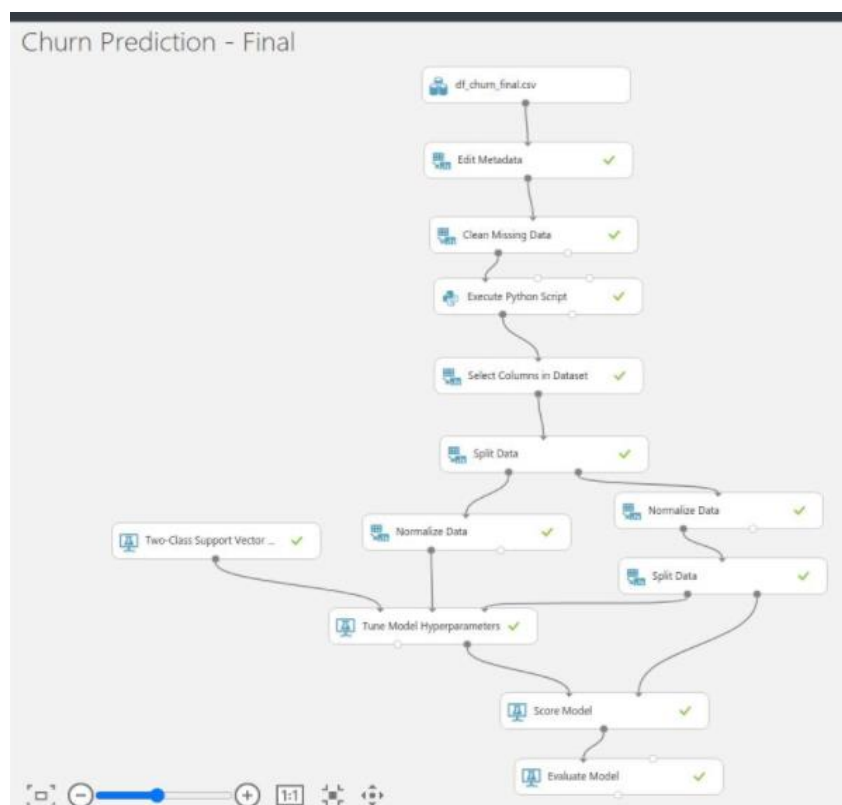
rows 34 columns 2

view as

Feature	Score
quantity_chronic_4	0.030596
quantity_chronic_5	0.025755
quantity_chronic_6	0.025755
quantity_chronic_7	0.025755
quantity_chronic_8	0.025755
quantity_chronic_9	0.025755
quantity_ethical_4	0.009252
quantity_generic_4	0.009216
quantity_ethical_5	0.001834
quantity_ethical_6	0.001834
quantity_ethical_7	0.001834
quantity_ethical_8	0.001834
quantity_ethical_9	0.001834
total_spend_bill	0.00157
num_drugs_bill	0.000839
quantity_acute_4	0.000515
quantity_generic_5	0.000515
quantity_generic_6	0.000515

Como segunda medida para melhorar o modelo final foi introduzida a operação de Tune Model Hyperparameters com o objetivo de encontrar uma melhor combinação de parâmetros.

Imagem 22 – Azure Machine Learning Tune Model Hyperparameters



O processo foi executado com os parâmetros apresentados em baixo:

Imagem 23 – Azure Machine Learning Parâmetros para Tune Model Parameter

Two-Class Support Vector Mac...

Create trainer mode  
 Parameter Range

Number of iterations  
☐ Use Range Builder  
 1, 10, 100

Lambda  
☐ Use Range Builder  
 0.00001, 0.0001, 0.001, 0.01, 0.1

☐ Normalize features

☐ Project to the unit-sp...

Random number seed  
 123

☒ Allow unknown categ...

Os parâmetros aconselhados pelo processo são os apresentados em baixo

Imagem 24 – Azure Machine Learning Tune Model Hyperparameters- Resultados

Churn Prediction - Final > Tune Model Hyperparameters > Trained best model

### Support Vector Machine Classifier

#### Settings

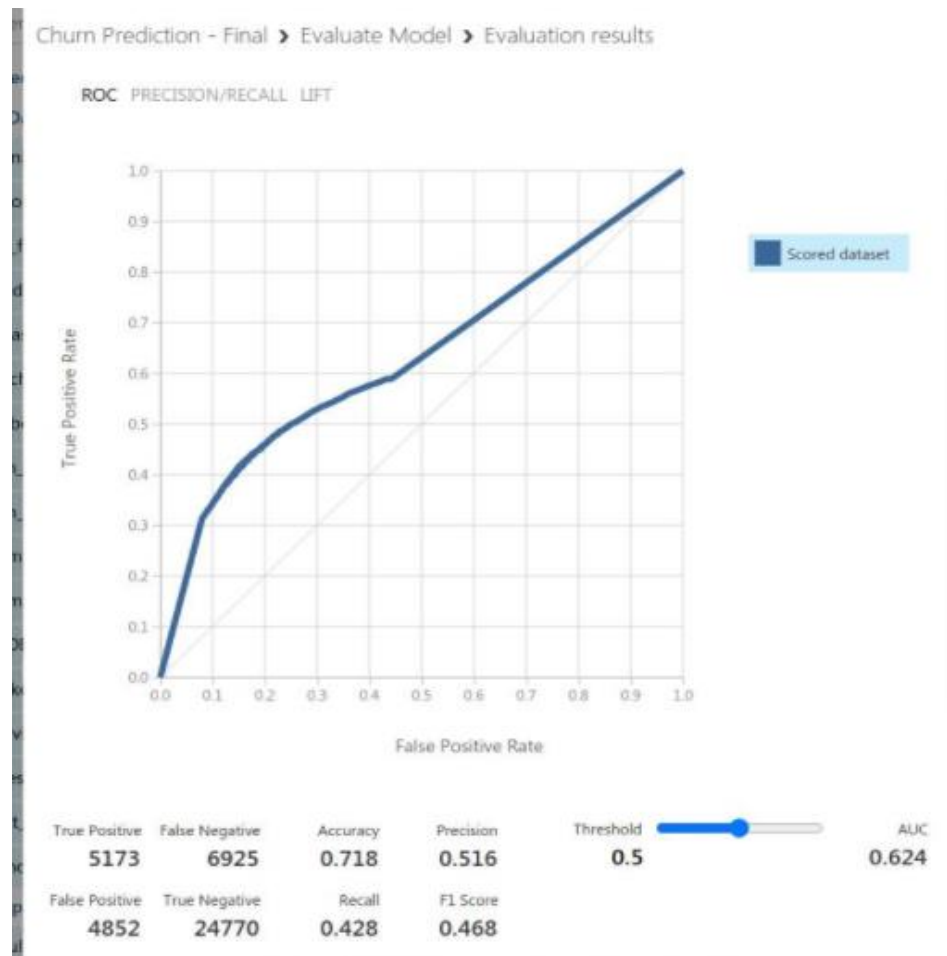
Setting	Value
Num Iterations	1
Lambda	0.0271752421
Normalize Features	False
Perform Projection	False
Allow Unknown Levels	True
Random Number Seed	123

#### Feature Weights

Feature	Weight
Bias	-1.00019
quantity_chronic_4	0.00594397
quantity_chronic_5	0.00548083
quantity_chronic_6	0.00548083
quantity_chronic_7	0.00548083
quantity_chronic_8	0.00548083
quantity_chronic_9	0.00548083
total_spend_bill	0.00387959
quantity_ethical_4	0.00306092
quantity_ethical_5	0.00291496

Após novo *Run* o modelo final apresenta os seguintes resultados e respetiva *Confusion Matrix*

Imagem 25 – Azure Machine Learning – Modelo Final Resultados





Por último também foi executada uma operação de Cross Validation com os resultados que podem ser vistos na imagem seguinte. Esta operação foi feita com o objetivo de verificar como generalizaria o modelo com novos dados, evitando que os resultados do nosso modelo estejam afetados por algum tipo de bias que condicione o resultado.

## Imagem 26 – Azure Machine Learning – Modelo Final – Cross Validation

Churn Prediction - Final > Cross Validate Model > Evaluation results by fold

rows 12 columns 10

view as  

	Fold Number	Number of examples in fold	Model	Accuracy	Precision	Recall	F-Score	AUC	Average Log Loss	Training Log Loss
0		27814	SVM (Pegasos-Linear)	0.737686	0.657945	0.173744	0.274896	0.632309	0.758099	-26.623776
1		27813	SVM (Pegasos-Linear)	0.740913	0.669586	0.174111	0.276361	0.626091	0.735232	-23.189948
2		27813	SVM (Pegasos-Linear)	0.733686	0.704597	0.11761	0.201574	0.617808	0.571381	4.512493
3		27813	SVM (Pegasos-Linear)	0.737677	0.676739	0.17565	0.278909	0.624465	0.886386	-47.462208
4		27814	SVM (Pegasos-Linear)	0.738082	0.665072	0.174295	0.276205	0.629293	0.754045	-25.843185
5		27814	SVM (Pegasos-Linear)	0.740886	0.666504	0.173257	0.275023	0.626705	0.743388	-24.648722
6		27813	SVM (Pegasos-Linear)	0.740373	0.694033	0.180981	0.287096	0.625807	0.858646	-42.839634
7		27814	SVM (Pegasos-Linear)	0.735313	0.685523	0.175762	0.279789	0.628955	0.739572	-22.367042
8		27814	SVM (Pegasos-Linear)	0.737722	0.683613	0.172285	0.275211	0.628007	0.742321	-23.45715
9		27814	SVM (Pegasos-Linear)	0.737255	0.666991	0.171607	0.272981	0.627636	0.75586	-26.008556
Mean		278136	SVM (Pegasos-Linear)	0.737959	0.67706	0.16893	0.269804	0.626707	0.754493	-25.792773
Standard Deviation		278136	SVM (Pegasos-Linear)	0.002336	0.014677	0.018217	0.024298	0.003816	0.083436	13.762199

## Conclusões

Como conclusões, e de uma forma sumária, podemos afirmar que o modelo tende a melhor performance com features mais atuais, uma alternativa seria utilizar modelos diferentes ou até mesmo uma rede neural LSTM onde seria possível utilizar a componente tempo.

Uma análise global do problema leva-nos a valorizar muito o indicador de Precision ( $\text{Precision} = \text{TP} / (\text{TP} + \text{FP})$ ) já que a existência de bastantes Falsos Positivos é mais penalizadora para o negócio, interferindo em eventuais orçamentos de vendas e orçamentos financeiros da empresa