Guilherme Moita Pontes

Configuração Automática de Dados no ANSA
Implementação de Métodos de Interpretação de Bases de Dados para uma Interface de Linguagem Natural para Pesquisas Analíticas

1 2 9 0

UNIVERSIDADE D
COIMBRA

Guilherme Moita Pontes

CONFIGURAÇÃO AUTOMÁTICA DE DADOS NO ANSA

Implementação de Métodos de Interpretação de Bases de Dados para uma Interface de Linguagem Natural para Pesquisas Analíticas

VOLUME 1

Relatório de Estágio no âmbito do Mestrado em Engenharia Informática, especialização em Sistemas Inteligentes orientada pelo Professor Doutor Carlos Bento da Universidade de Coimbra e pelos Eng. Tiago Baptista e Rui Lopes da Critical Software SA. e apresentada à Faculdade de Ciências e Tecnologia / Departamento de Engenharia Informática.

Setembro de 2021

Faculdade de Ciências e Tecnologia
Departamento de Engenharia Informática

Configuração Automática de Dados no ANSA

Implementação de Métodos de Interpretação de Bases de Dados para uma Interface de Linguagem Natural para Pesquisas Analíticas

Guilherme Moita Pontes

Relatório de Estágio no âmbito do Mestrado em Engenharia Informática, especialização em Sistemas Inteligentes orientada pelo Professor Doutor Carlos Bento da Universidade de Coimbra e pelos Eng. Tiago Baptista e Rui Lopes da Critical Software SA. e apresentada à Faculdade de Ciências e Tecnologia / Departamento de Engenharia Informática.

Setembro de 2021

1 2 9 0

UNIVERSIDADE D
COIMBRA

Resumo

Este relatório documenta as atividades realizadas relativas ao estágio curricular na Critical Software SA (CSW) no âmbito do Mestrado em Engenharia Informática da Faculdade de Ciências e Tecnologias da Universidade de Coimbra.
O principal objetivo deste trabalho é melhorar a configuração de um programa desenvolvido pela CSW, o ANSA. O ANSA é uma interface de linguagem natural para pesquisas analíticas, ou seja, é um programa que permite fazer perguntas em linguagens como português ou inglês acerca de uma base de dados onde as respostas são depois dadas no formato mais adequado, seja em texto simples ou sob a forma de gráficos. Para que isto seja possível existe a necessidade de configurar a base de dados de forma a ser interpretável pelo sistema. O principal objetivo deste estágio é então automatizar este processo reduzindo o tempo necessário e aumentando a facilidade de adição de novos conjuntos de dados ao ANSA. Para tal, será criada uma plataforma web nomeada "*Back Office*" onde será possível gerar configurações. Estas configurações devem classificar a base de dados fornecida quanto às categorias dos seus dados (dados de pessoas, organizações, datas, entre outros), quanto ao tipo de dados (inteiros, decimais, letras, entre outros) e sugerir o nome a apresentar na interface para cada coluna.
Para além de visualizar as sugestões de configuração, esta ferramenta também deve permitir alterar definições do ANSA, visualizar estatísticas relativas ao comportamento dos utilizadores, entre outros requisitos definidos neste relatório.

Palavras-Chave

Interface de Linguagem Natural para Pesquisas Analíticas, Processamento de Linguagem Natural, Extração de Informação, Reconhecimento de Entidades Mencionadas, Classificação de dados

Abstract

This document reports the work done regarding the curricular internship at Critical Software (CSW) within the scope of the Master's in Informatics Engineering at the Faculty of Sciences and Technologies of the University of Coimbra.

The main goal of this work is to improve the configuration of a software developed by CSW, named ANSA. ANSA is a natural language interface for search analytics, i. e., a program that allows a user to ask questions about a given database in languages like English or Portuguese and get answers in the most appropriate format, be it text or graphs. To make this possible, there is a need to configure the database, making it interpretable. Therefore, the main focus of the internship is to automate this process, reducing the configuration time and making it easier to add new databases to ANSA. To do so, a web platform called "back office" will be created where these configurations will be created. The configurations must classify the given data by its category (personal, organizational, dates, among others), by its data type (integers, decimal numbers, strings, among others) and suggest the names to be displayed on the interface for each column.

Besides creating and visualizing configuration, this tool should allow the changing of ANSA's settings and show statistics that help the business administrator analyze how the users behave in the platform, among other requirements defined in this document.

Keywords

Natural Language Interface for Search Analytics, Natural Language Processing, Information Extraction, Named Entity Recognition, Data Classification

Agradecimentos

Quero agradecer aos meus orientadores, ao Professor Doutor Carlos Bento, ao Engenheiro Tiago Batista e ao Engenheiro Rui Lopes, por todo o acompanhamento, apoio prestado e disponibilidade neste ano atípico que obrigou a mudar a forma como trabalhamos. Aprendi muito convosco e tenho muito mais a aprender.
À equipa da Critical Software SA pela oportunidade de integração e pela forma positiva e prestabilidade com que sempre fui recebido.
Aos meus pais e familiares, por me darem a oportunidade de seguir a educação que quero, por estarem sempre do meu lado a incentivarem-me e a desejarem o melhor para mim.
A todos os meus amigos em que, durante o total dos cinco anos académicos, partilhamos as partes boas, também as partes más, criámos muitas memórias que espero nunca perder e acima de tudo por celebrarem a vida comigo.

Índice

Capítulo 1 Introdução 1

1.1 Contextualização 1

1.2 Objetivos e critérios de sucesso 2

1.3 Estrutura do documento 2

Capítulo 1 – Introdução 2
Capítulo 2 – Conhecimento prévio 2
Capítulo 3 – Estado da arte 3
Capítulo 4 – Plataforma ANSA 3
Capítulo 5 – Abordagem 3
Capítulo 6 – Metodologia e planeamento 3
Capítulo 7 – Implementação 3
Capítulo 8 – Testes 3
Capítulo 9 – Conclusão 3

Capítulo 2 Conhecimento prévio 5

2.1 Processamento de Linguagem Natural 5

2.1.1. Abordagens em linguística 5

2.1.2. Abordagens em PLN 6

Tokenização 6
Análise lexical 8
Análise sintática 9
Análise semântica 10

2.2 Reconhecimento de entidades mencionadas 10

2.2.1. Técnicas manuais 11

Baseadas em regras 11
Baseadas em dicionários 11

2.2.2. Técnicas de aprendizagem computacional 12

Modelos Ocultos de Markov (MOM) 12
Modelos Markov de Entropia Máxima (MMEM) 13
Campos Aleatórios Condicionais (CAC) 13
Support Vector Machines (SVM) 13

2.2.3. Técnicas de ontologia 14

Capítulo 3 Estado da arte 15

3.1 Ferramentas de REM 15

3.2 Produtos atualmente no mercado 16

SQL Server 16
ApexSQL Mask 17
Netwrix Data Classification 18
Resumo 19

Capítulo 4 Plataforma ANSA 20

Capítulo 5 Abordagem 22

5.1 Requisitos 22

Requisitos funcionais 22
Requisitos não funcionais 25
Análise de riscos 26

Capítulo 6 Metodologia e planeamento 29

6.1 Metodologia de trabalho 29

6.2 Planeamento 30

Capítulo 7 Implementação 33

7.1 Modificações no ANSA 33

7.2 Configurações 36

Configuração automática 37

Configuração manual 39

7.3 Classificação 40

Objetivo 40

Conjunto de dados 41

Pré-processamento dos dados 43

Classificadores 45

7.4 Acesso à API 48

Capítulo 8 Testes 51

8.1 Classificação 51

8.2 Configuração automática 57

Capítulo 9 Conclusão 60

Referências 61

Acrónimos

Abreviaturas	Descrições
BOW	*Bag of Words*
BD	Base de Dados
CCPA	*California Consumer Privacy Act*
CAC	Campos Aleatórios Condicionais
MUC	Conferência de Entendimento de Mensagens
CSW	Critical Software SA.
DEI	Departamento de Engenharia Informática
GDPR	*General Data Protection Regulation*
GNU	Licença Pública Geral
HTTP	*Hypertext Transfer Protocol*
IA	Inteligência Artificial
ISO	Organização Internacional de Normalização
KNN	*K-Nearest Neighbor*
LDA	*Linear Discriminant Analisys*
LGBM	*Light Gradient Boosting Machines*
MMEM	Modelos Markov de Entropia Máxima
MOM	Modelos Ocultos de Markov
NLTK	*Natural Language Tool Kit*
OMS	Organização Mundial de Saúde
PCA	Análise de componentes principais
POS	*Part of Speech*
PLN	Processamento de Linguagem Natural
REM	Reconhecimento de Entidades Mencionadas
RF	*Random Forests*
RNF	Requisito Não Funcional
SGBD	Sistemas de Gestão de Bases de Dados
SQL	*Structured Query Language*
SVM	Support Vector Machines
UC	Universidade de Coimbra

Lista de Figuras

Figura 1 - Exemplo de tokenização usando como delimitador o espaço em branco. 6

Figura 2 - Codificação da frase "A Maria apanhou o metro." por one-hot encoding..... 7

Figura 3 - Codificação de frases por *Bag of Words* 7

Figura 4 - Exemplo da distribuição de palavras no espaço vetorial usando *word embeddings* 8

Figura 5 - Exemplo de *POS tagging* 9

Figura 6 - Árvore sintática para a frase "O jogo acabou empatado." 9

Figura 7 - Exemplo de REM na frase "O João estuda na UC desde 2018." 10

Figura 8 - Exemplo de uma expressão regular para encontrar cartões de crédito Visa 11

Figura 9 - Exemplo de um hiperplano que separa as duas classes..... 13

Figura 10 - Interface da ferramenta de deteção e classificação de dados do SQL Server 16

Figura 11 - Interface do programa ApexSQL Mask 17

Figura 12 - Interface do programa Netwrix Data Classification 18

Figura 13 - Exemplo de um formato de resposta dada pelo ANSA 20

Figura 14 - Resposta dada pelo ANSA no formato de *pie-chart* 21

Figura 15 - Diagrama de Gantt inicial relativo ao primeiro semestre 30

Figura 16 - Diagrama de Gantt final relativo ao primeiro semestre..... 30

Figura 17 - Diagrama de Gantt inicial relativo ao segundo semestre 31

Figura 18 - Diagrama de Gantt final relativo ao segundo semestre 31

Figura 19 - Exemplo de uma pesquisa no ANSA utilizando os distritos portugueses..... 33

Figura 20 - Exemplo de pesquisa que mostra no mapa o número de mortos por Covid19 em cada país..... 34

Figura 21 - Exemplo de pesquisa que mostra o número de novos casos de Covid19 em Portugal no dia 1 de julho de 2021 34

Figura 22 - Exemplo de uma pesquisa que mostra a evolução da abstenção nas eleições legislativas portuguesas desde 1976 até 2011. 35

Figura 23 - Exemplo de pesquisa que apresenta o Primeiro-Ministro eleito em 2011. 35

Figura 24 - Esquema da configuração *Docker* utilizada para desenvolvimento 36

Figura 25 - Diagrama de fluxo com as operações executadas durante a configuração automática 38

Figura 26 - Histograma com a distribuição da classe "anotações e tipo" 42

Figura 27 - Histograma com a distribuição da classe "tipo" 42

Figura 28 - Histograma com a distribuição da classe "anotações e tipo" 43

Figura 29 - Tradução dos tipos de dados na BD para os tipos de dados no ANSA 45

Figura 30 - Diagrama de fluxo com as regras de classificação de anotações 46

Figura 31 - Divisão dos métodos "ensemble" 47

Figura 32 - Fórmula exatidão 53

Figura 33 - Fórmula precisão 53

Figura 34 - Fórmula abrangência 53

Figura 35 - Fórmula *f1-score* 53

Figura 36 - Histograma com os melhores resultados obtidos na abordagem de classificação única. 54

Figura 37 - Tempos de treino relativos à abordagem de classificação única 54

Figura 38 - Histograma com os melhores resultados obtidos na abordagem de classificação em série. 55

Figura 39 - Tempos de treino relativos à abordagem de classificação em série 56

Figura 40 – Interface do ANSA com pesquisas realizadas sobre a configuração de eleições legislativas portuguesas. 58

Figura 41 – Interface do ANSA testando a anotação de países numa configuração com dados de vendas. 58

Lista de Tabelas

Tabela 1. Tabela comparativa das ferramentas de REM.....15

Tabela 2. Tabela comparativa dos produtos disponíveis no mercado. 19

Tabela 3. Requisitos funcionais de alto nível 23

Tabela 4. Requisitos Funcionais de RF-1: Extração e interpretação automática de informação. 23

Tabela 5. Requisitos Funcionais de RF-2: *Back Office*. 24

Tabela 6. Requisitos Funcionais de RF-3: Painel de controlo.....24

Tabela 7. Requisitos não funcionais 25

Tabela 8. Risco 1 - Integração e comunicação com a equipa.....26

Tabela 9. Risco 2 - Familiarização com as ferramentas.....26

Tabela 10. Risco 3 - Configuração automática não permite as melhorias esperadas.....27

Tabela 11. Risco 4 – REM não permite classificações corretas.....27

Tabela 12. Risco 5 - Tarefas não completadas dentro dos prazos 27

Tabela 13. Risco 6 - Tempo para completar todos os requisitos não é suficiente.....27

Tabela 14. Risco 7 - Familiarização demorada com a implementação atual do ANSA.....28

Tabela 15. Risco 8 - Bibliotecas disponíveis em Java não satisfazem as necessidades de implementação.....28

Tabela 16. Risco 9 - Falta de experiência em desenvolvimento web leva a problemas de performance ou segurança 28

Tabela 17. Componentes de uma configuração 37

Tabela 18. Descrição do conjunto de dados.....41

Tabela 19. *Endpoints relativos à configuração*. 49

Tabela 20. Tabela com todos os valores testados.....52

Tabela 21. Tabela com os parâmetros de melhor performance para as RF na abordagem de classificação única.....55

Tabela 22. Tabela com os parâmetros de melhor performance para as RF na abordagem de classificação em série.....56

Tabela 23. Resultados dos testes às configurações 59

Capítulo 1
Introdução

Este documento relata as atividades realizadas durante o estágio curricular na Critical Software SA (CSW) no âmbito do Mestrado em Engenharia Informática da Faculdade de Ciências e Tecnologias da Universidade de Coimbra, cujo objetivo é melhorar a configuração de um programa proprietário, o ANSA.
A CSW é uma empresa com sede em Coimbra e de implantação internacional especializada em oferecer soluções, serviços e tecnologias fiáveis para sistemas de informação de missão crítica em vários tipos de mercados. Atualmente, está a desenvolver uma plataforma de linguagem natural para pesquisas analíticas, o ANSA. Este programa tem como objetivo permitir ao utilizador fazer pesquisas em linguagem natural relativas a uma base de dados e receber os resultados no formato mais adequado (gráfico, texto, entre outros). Deste modo, qualquer pessoa pode retirar informações de uma determinada base de dados utilizando a sua linguagem natural, como o português ou inglês, sem ter de recorrer a linguagens formais, como SQL. Com estas interfaces de pesquisa em linguagem natural, o processo de obter informação de bases de dados torna-se mais acessível já que não é necessário aprender linguagens formais, tradicionalmente usadas para realizar tais pesquisas.

1.1 Contextualização

Tradicionalmente, para um empresário tomar uma decisão informada acerca da sua empresa é necessária a intervenção de duas entidades até a informação estar pronta para ser interpretada. Primeiro, um técnico conhecedor de linguagens formais, como SQL, retira a informação da base de dados da empresa e, em segundo, um analista forma relatórios tornando essa informação facilmente interpretável. As ferramentas de análise de dados desenvolvidas nos anos 80 ainda são usadas atualmente e, nos anos 90 e início dos anos 2000, com a mudança dos dados para a *cloud* (Setlur et al., 2018) o armazenamento de dados começou a crescer exponencialmente até chegarmos à era da *Big Data* em que vivemos hoje. Apesar das primeiras interfaces para visualização de dados começarem a aparecer nessa altura, a metodologia usada era a mesma, obrigando à intervenção de uma equipa informática e de uma equipa de analistas e fazendo com que um relatório demorasse em média 4 a 7 dias até ser produzido (Setlur et al., 2018).
É aqui que entram as interfaces de pesquisa em bases de dados através de linguagem natural que permitem obter dados facilmente interpretáveis fazendo pesquisas em linguagens como o português ou inglês, como se de um motor de busca comum se tratasse. Desta forma, um utilizador não-técnico pode obter os dados que lhe são relevantes sem necessitar de recorrer a equipas de informáticos e analistas, tudo numa questão de segundos.

Para que toda esta automação seja possível em breves instantes é necessário transformar as bases de dados originais em meta dados facilmente interpretados por estes sistemas seguindo uma metodologia e estrutura específicas. Sendo o objetivo destas plataformas a agilização da obtenção de dados empresariais e criação de relatórios, não faz sentido que a sua adaptação a novos dados seja demorada e complexa. É aqui que entra o âmbito deste estágio, que procura criar uma ferramenta capaz de interpretar e configurar automaticamente novos conjuntos de dados para o ANSA, a interface de pesquisa através de linguagem natural da CSW.

1.2 Objetivos e critérios de sucesso

Atualmente a configuração de novas bases de dados para a plataforma ANSA é um processo demorado, manual e bastante dependente do administrador do sistema e do seu conhecimento técnico. O configurador recorre a scripts e à sua interpretação manual da base de dados de forma a integrá-la com a plataforma. Assim, o trabalho realizado durante o estágio tem como objetivo diminuir o tempo e a complexidade associados a este processo, criando um *back office* que se tornará na ferramenta de apoio à configuração de novas bases de dados. Para além de apresentar sugestões de configuração, esta ferramenta também deve permitir alterar definições do ANSA e visualizar estatísticas relativas ao comportamento dos utilizadores.
Com estes objetivos, o estágio realizado na CSW é avaliado segundo os critérios descritos de seguida:

- Ter uma ferramenta que, dada uma base de dados, sugira a configuração a utilizar no ANSA: testar várias soluções baseadas em sistemas de regras e em sistemas de modelos que permitam, dada uma base de dados, gerar sugestões de integração com o ANSA classificando o conteúdo dos dados em categorias (por exemplo pessoais, financeiros, entre outros) e no seu tipo (por exemplo números, letras, entre outros).

- Criação de uma plataforma de apoio ao ANSA nomeada *Back Office*: esta plataforma apresentará ao utilizador os resultados da configuração automática descrita, permitirá fazer alterações manuais à configuração da base de dados, ao aspeto visual e às definições de idioma. Por último, apresentará informações estatísticas que permitam ao administrador perceber o comportamento dos utilizadores na plataforma ANSA.

1.3 Estrutura do documento

Este relatório encontra-se dividido por capítulos da forma a seguir descrita.

Capítulo 1 – Introdução

Capítulo atual, onde é introduzido e contextualizado o trabalho a realizar durante o estágio e enunciados os seus objetivos.

Capítulo 2 – Conhecimento prévio

Neste capítulo são introduzidos os conceitos teóricos sobre Processamento de Linguagem Natural (PLN) que tornam possível a implementação do trabalho a realizar

durante o estágio. De seguida, são apresentadas as tarefas que tornam possível a interpretação dos dados, nomeadamente o Reconhecimento de Entidades Mencionadas (REM).

Capítulo 3 – Estado da arte

Neste capítulo analisamos as soluções que já existem no mercado. Numa primeira fase olhamos para as ferramentas de REM que podem ser usadas neste trabalho. Por último, olhamos para produtos existentes no mercado que interpretam automaticamente bases de dados.

Capítulo 4 – Plataforma ANSA

Neste capítulo é feita uma breve introdução do programa ANSA e é demonstrado o seu funcionamento com exemplos da sua interface. Esta explicação tem como objetivo melhorar a interpretação dos requisitos enunciados no capítulo seguinte.

Capítulo 5 – Abordagem

Neste capítulo são detalhadas as funcionalidades a implementar bem como o seu funcionamento e objetivos, apresentando os requisitos funcionais e não funcionais. É feita uma análise de riscos com os correspondentes planos de mitigação e, por último, são apresentadas as especificações técnicas.

Capítulo 6 – Metodologia e planeamento

Neste capítulo é explicada a forma como o trabalho se realizou ao longo do ano, como foi planeado e as alterações que esse plano sofreu.

Capítulo 7 – Implementação

Aqui são enunciadas todas as tarefas realizadas que permitiram chegar ao objetivo do estágio. Ao longo do processo, é explicado o raciocínio por detrás das decisões.

Capítulo 8 – Testes

Neste capítulo é apresentada a metodologia e os resultados dos testes aos vários classificadores implementados e também ao protótipo final.

Capítulo 9 – Conclusão

Neste capítulo é apresentado um resumo do trabalho realizado durante o estágio. É feito um balanço da experiência e são abordados os planos futuros.

Capítulo 2
Conhecimento prévio

Neste capítulo são apresentadas noções importantes que permitem a implementação da componente mais exploratória deste estágio: a interpretação automática de informação proveniente de bases de dados de forma a criar uma configuração para o ANSA.
Quando falamos de interpretação de informação em linguagem natural estamos a referir-nos ao Processamento de Linguagem Natural (PLN) detalhado na secção 2.1.
Na secção 2.2 aprofundamos a temática do PLN mais útil ao âmbito deste estágio: o Reconhecimento de Entidades Mencionadas (REM), descrevendo as várias abordagens à extração e classificação de entidades.

2.1 Processamento de Linguagem Natural

O PLN é uma área da Inteligência Artificial que tem como objetivo permitir à tecnologia ler, interpretar e deduzir o significado da linguagem dos seres humanos. Apesar de ser uma área recente e bastante ativa em investigação já nos traz funcionalidades como assistentes virtuais (como a Siri ou o Google Assistente), tradução automática de documentos e imagens, resumo automático de textos, perguntas e respostas feitas em linguagem natural, entre outros…
Referimo-nos a linguagem natural como a linguagem usada por humanos, como o português, inglês, etc. Por outro lado, linguagem formal diz respeito às típicas linguagens de programação, como Java, SQL, C, entre outras.

2.1.1. Abordagens em linguística

A linguística é o estudo da linguagem e foi esta área científica que inspirou várias técnicas usadas no PLN. A linguística analisa a linguagem em cada um dos níveis apresentados de seguida:

- **Fonética e fonologia:** fonética é o estudo dos sons humanos e fonologia é a classificação desses sons de acordo com a sua produção, transmissão e perceção.
- **Morfologia:** corresponde à análise lexical da palavra. Estuda como as palavras são formadas e classifica-as em classes, como nomes, verbos, etc. Esta classificação é muito útil em PLN, denominada de *Part of Speech* (POS).
- **Sintaxe:** define como as palavras se relacionam quando são enunciadas ou escritas em frases, orações e períodos. Aqui é estudada a estrutura da frase e as relações entre os seus elementos.
- **Semântica:** atua ao nível do significado. A semântica lexical reflete o significado individual das palavras enquanto que a semântica frásica reflete o significado de toda a frase, resultante da interação entre os seus elementos.

- **Pragmática:** neste nível é analisado o significado que não é transmitido nos níveis anteriores. Aqui é tido em conta o contexto extralinguístico como a situação, discurso e emoção.

Visto o PLN partilhar com a linguística o objetivo de analisar e interpretar a linguagem natural, os níveis descritos podem também ser encontrados nas abordagens computacionais. Esta junção entre inteligência artificial e linguística não só beneficiou a área de PLN como também a área da linguística permitindo compreender melhor a história e formação das linguagens, estimar divergências temporais entre famílias de linguagens e facilitar o estudo comparativo entre as mesmas (Sharma, 2010).

2.1.2. Abordagens em PLN

Como referido anteriormente, algumas das fases de análise aplicadas em linguística são também úteis ao PLN. Uma típica abordagem passa por uma fase de tokenização, análise lexical, análise sintática e análise semântica. No entanto, estas tarefas estão dependentes dos objetivos de cada trabalho, podendo ser ou não necessárias ou até mesmo diferentes.

Tokenização

Esta fase ocorre, tipicamente, durante o pré-processamento dos dados. Em PLN o texto é encarado como uma sequência de palavras. A tokenização tem como objetivo dividir as sequências originais de texto em peças mais pequenas nomeadas *tokens* para mais tarde facilitar o seu processamento semântico. Esta divisão pode ser feita escolhendo delimitadores como espaços e pontuação, por pares de caracteres mais frequentes, entre outros. É apresentado um exemplo ilustrativo de seguida.

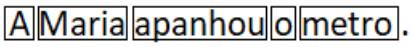

Figura 1 - Exemplo de tokenização usando como delimitador o espaço em branco.

Os vários *tokens* criados representam um vocabulário que para ser computacionalmente processado deve ser representado de forma numérica. Esta representação numérica tem o nome de vectorização e pode seguir diferentes lógicas consoante o tipo de problema que queremos resolver. Identificam-se dois grupos:

1. **Representações vetoriais esparsas**

Estas representações foram as primeiras a surgir e são as mais simples. A primeira vectorização de texto a ser utilizada foi o *one-hot encoding* e ainda é usada atualmente para algumas tarefas. Nesta abordagem cada palavra encontrada é convertida num vetor com o tamanho de todo o vocabulário, em que o índice correspondente à palavra em questão toma o valor 1 (*one-hot*) e todos os outros índices tomam o valor 0. Exemplifiquemos com a frase anterior:

tokens a codificar ↓

	A	Maria	apanhou	o	mais	um	metro
A	1	0	0	0	0	0	0
Maria	0	1	0	0	0	0	0
apanhou	0	0	1	0	0	0	0
o	0	0	0	1	0	0	0
metro	0	0	0	0	0	0	1

← vocabulário completo

Figura 2 - Codificação da frase "A Maria apanhou o metro." por one-hot encoding

Apesar do conceito ser simples de compreender e implementar, o resultado é uma matriz esparsa em que grande parte dos valores é zero. Isto leva a que o crescimento do vocabulário aumente rapidamente o espaço de armazenamento necessário. No entanto, a maior desvantagem desta abordagem é o facto de apenas manter informação acerca da presença/ausência das palavras, ignorando contexto, ordem, significado ou semelhança entre palavras.

Se a codificação *one*-hot nos permite obter vetores para cada *token*, por outro lado, a abordagem *Bag of Words (BOW)* permite-nos obter vetores para cada frase. Nesta abordagem, o texto é convertido para um vetor, do tamanho do vocabulário, em que cada índice representa uma palavra e cada valor representa o número de ocorrências dessa palavra no texto. Exemplificando:

	o	a	João	Maria	apanhou	um	peixe	metro	de
O João apanhou um peixe de um metro. →	1	0	1	0	1	2	1	1	1
A Maria apanhou o metro. →	1	1	0	1	1	0	0	1	0

Figura 3 - Codificação de frases por *Bag of Words*

A intuição por trás da *BOW* é que são as próprias palavras que nos trazem a informação necessária à compreensão. Mas nem sempre é o caso já que a ordem e o contexto das palavras numa frase alteram o seu significado, como acontece com a palavra "metro" na Figura 3. Mais uma vez, também nesta abordagem não temos informação acerca do significado. Ainda assim, a informação captada por estas representações é suficiente para várias tarefas de PLN como análise de sentimentos.

2. Representações vetoriais densas

Ao contrário do que acontece nas representações vetoriais esparsas, nas representações vetoriais densas variáveis como significado, relações semânticas e os diferentes contextos são tidas em conta. Tudo isto é possível recorrendo a *word embeddings. Word embeddings* é o mapeamento de palavras para vetores de N dimensões (geralmente entre 50 e 300), em que cada uma destas dimensões tem um significado atribuído. A forma como a palavra se relaciona com esse significado define o valor que terá nessa dimensão. Por exemplo, sendo as palavras "homem" e "mulher" semelhantes no sentido em que ambas se referem a seres humanos, é de esperar que muitas das dimensões tenham valores semelhantes e que a maior diferença seja na

dimensão que representa o género. Assim, podemos esperar que palavras semelhantes estejam mais próximas no espaço vetorial, como exemplificado na Figura seguinte:

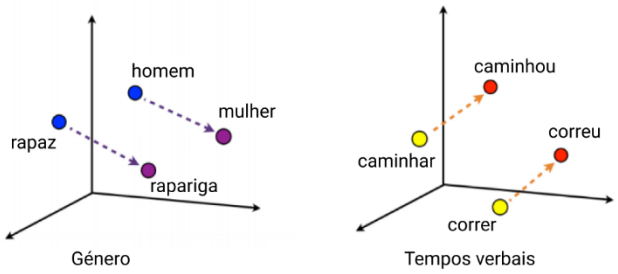

Figura 4 - Exemplo da distribuição de palavras no espaço vetorial usando *word embeddings*

Estes vetores são normalmente produzidos por redes neuronais treinadas em grandes quantidades de texto que procuram encontrar o valor correto para cada dimensão do vetor. O treino destes modelos é computacionalmente caro, mas existem vários modelos pré-treinados disponibilizados como o *Word2Vec* da Google e o *FastText* do Facebook.

Apesar da complexidade e tempo necessário à implementação, estas abordagens permitem captar relações semânticas e o contexto das palavras e suportam tarefas como tradução automática e respostas a perguntas em linguagem natural.

Em jeito de conclusão, o método escolhido para vectorização está dependente do tipo de tarefas a mais tarde desempenhar.

Análise lexical

Analogamente à linguística, nesta fase o PLN procura obter significado ao nível das palavras. Para tal, pode ser útil normalizar os *tokens* já obtidos, tipicamente, através de *stemming* ou *lematização.*

Stemming é o processo de remover ou substituir sufixos de uma palavra de forma a obter a sua raiz. Ou seja, são retirados os radicais ficando apenas com a raiz da palavra. Por exemplo, se considerarmos as palavras “beber” e “bebida” vemos que a sua raiz é “beb”. Apesar da rapidez de implementação de um algoritmo deste género, ficamos com uma palavra que não existe no dicionário e que pode não corresponder ao lema da palavra original. Outra desvantagem é o facto de perdermos informações importantes como saber se a palavra original era um nome ou um verbo.

Lematização é o processo substituir as palavras originais pelo seu lema. O lema é a forma gráfica de uma palavra que é usada como entrada de verbete num dicionário. Por exemplo, o lema de “tem” e de “tiver” é o verbo “ter”, o lema de “bom” e de “melhor” é o adjetivo “bom”. Apesar de ser um processo mais complexo que o *stemming*, nesta abordagem obtemos sempre palavras existentes no dicionário e preservamos a sua função sintática.

Dependendo dos objetivos da implementação podem ser benéficos outros métodos de normalização como a transformação de letras maiúsculas em minúsculas. Esta transformação pode ser aplicada apenas no início das frases ou apenas a títulos, deixando intactas as palavras a meio da frase, já que nestes casos a letra maiúscula indica-nos a presença de um nome próprio.

Por último, podemos acrescentar aos *tokens* obtidos informação gramatical acerca da(s) palavra(s) neles contida(s). Este processo tem o nome de *Part of speech (POS) tagging* e adiciona a cada *token* a função da(s) sua(s) palavra(s) numa frase, ou seja, se a palavra representa um nome, um verbo, entre outros… Este processo pode não ser tão simples como parece visto que uma mesma palavra pode ter funções diferentes consoante a frase e o contexto em que aparece. Para identificar qual a função mais provável é preciso ter em conta as características lexicais e sintáticas da palavra. Assim, costumam ser usados identificadores *POS* baseados em regras ou em métodos estatísticos treinados em dados anotados para calcular a probabilidade da palavra ter uma dada função, consoante o contexto.

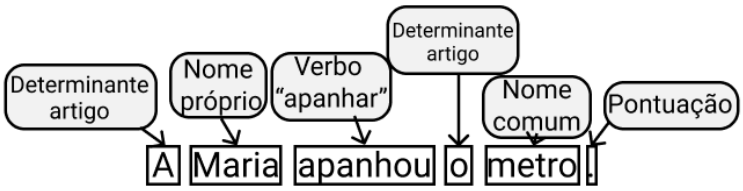

Figura 5 - Exemplo de *POS tagging*

Análise sintática

Contrariamente à análise lexical, a análise sintática é feita ao nível da frase e não apenas a cada palavra individualmente. Nesta fase é analisada a estrutura do texto garantido que segue as regras de uma gramática formal. É também possível ficar a conhecer as dependências e relações entre palavras de cada frase.

Esta análise é feita, tipicamente, recorrendo a gramáticas livres de contexto e tem como objetivo delimitar a hierarquia sintática para depois ser usada na análise semântica. Esta hierarquia é representada através de árvores sintáticas que permitem visualizar facilmente relações entre as palavras. São iniciadas por uma raiz que expande para símbolos não terminais quando encontra uma regra de produção e no final chega-se às folhas das árvores, correspondendo aos símbolos terminais que representam as palavras encontradas na frase. A Figura 6 exemplifica uma árvore de sintaxe para a frase "O jogo acabou empatado.".

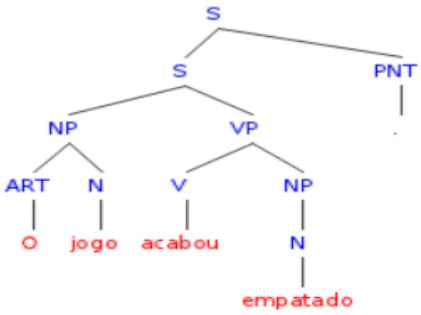

Figura 6 - Árvore sintática para a frase "O jogo acabou empatado."[1]

[1] Fonte: LX-Center Constituency Parser

Análise semântica

O objetivo da análise semântica é retirar significado concreto e preciso do texto interpretado. Enquanto que a análise lexical descrita anteriormente procura o significado ao nível das palavras, a análise semântica procura o significado em pedaços de texto maiores. Se para nós, humanos, retirar o significado da linguagem é intuitivo para a máquina esta é uma tarefa complicada pela ambiguidade, contexto e figuras de estilo, como o sarcasmo, inerentes à linguagem natural.
Podemos considerar quatro pilares construtores do significado:

- **Entidades:** representam pessoas, localizações, etc. Ex: João, Coimbra;
- **Conceitos:** representam a categoria a que as entidades pertencem. Ex: pessoa, instituição;
- **Relações:** identificam as conexões entre entidades e conceitos. Ex: O João é uma pessoa;
- **Predicados:** representam as estruturas verbais. Apresentam-nos a relação entre o sujeito e as ações e permitem identificar as orações verbais.

Juntando as entidades, conceitos, relações e predicados pretende-se que a máquina seja capaz de interpretar uma situação. Para captar estas 4 componentes usam-se técnicas como o reconhecimento de entidades mencionadas, extração de relações, análise de sentimentos, desambiguação de palavras, entre outras.

2.2 Reconhecimento de entidades mencionadas

O REM, também referido como extração de entidades, é um dos primeiros passos a tomar em direção à extração de informação em PLN. É a tarefa de identificar e categorizar informações chave (entidades) num texto. O conceito "entidade mencionada" foi criado em 1996, na Conferência de Entendimento de Mensagens - 6 (MUC-6) (Grishman & Sundheim, 1996) e referia-se a nomes de pessoas, localizações, organizações e expressões numéricas como valores monetários e datas. Por exemplo, um modelo de REM deve identificar a palavra "João" e incluí-la na categoria "Pessoa".

Figura 7 - Exemplo de REM na frase "O João estuda na UC desde 2018."

Apesar dos tipos de entidades referidos anteriormente, as categorias a utilizar em cada implementação estão dependentes do tipo de texto a interpretar e do seu conteúdo. Por exemplo, em textos científicos da área da farmacêutica faz sentido incluir uma categoria "medicamento" ou "sequência DNA". Esta capacidade torna o REM útil em várias áreas como a dos motores de busca, melhorando a relevância dos resultados e ajudando a resumir os conteúdos encontrados; no apoio ao cliente, permitindo categorizar pedidos, reclamações e questões e filtrar por palavras chave; na saúde, extraindo informação essencial dos relatórios de condição clínica de cada paciente; nos recursos humanos permitindo resumir os currículos dos vários candidatos, entre outros. Para que tudo isto seja possível, qualquer abordagem de REM obriga a duas tarefas:

1. Detetar a entidade mencionada.
2. Classificar a entidade detetada.

De seguida, serão apresentados os tipos de abordagens a seguir para o REM que podem ser divididas em técnicas manuais, técnicas baseadas em aprendizagem computacional e técnicas baseadas em ontologia.

2.2.1. Técnicas manuais

Estas técnicas são codificadas à mão e dividem-se em duas abordagens: baseadas em regras, onde padrões são usados para identificar entidades e baseadas em dicionários, onde os *tokens* são comparados com vocábulos de um dicionário de entidades.

Baseadas em regras

Estas abordagens criam regras usando padrões, tais como expressões regulares, de forma a encontrar *tokens* que correspondam e ativem essas regras, identificando-os assim como entidades. Estas regras são normalmente baseadas na gramática da linguagem natural em questão e vão desde as mais simples até às mais complexas. O primeiro conjunto de regras nesta área foi criado por Lisa Rau (Rau, 1991) e tinha como objetivo identificar nomes de empresas recorrendo a regras como a palavra ser iniciada por letra maiúscula e a deteção de sufixos como "Lda.", "S.A.", entre outros. Outro exemplo de padrão pelo qual podemos procurar é o prefixo "Sr(a)." seguido de uma ou mais palavras iniciadas por letra maiúscula, indicando a menção de uma pessoa (Mikheev et al., 1999).
Estes sistemas são bastante úteis para encontrar entidades que seguem obrigatoriamente uma determinada estrutura. Exemplos disso são números de cartões de cidadão, códigos postais ou número de cartões de crédito, como é apresentado na Figura 8.

```
^4[0-9]{12}(?:[0-9]{3})?$
```

Figura 8 - Exemplo de uma expressão regular para encontrar cartões de crédito Visa

Apesar de manualmente codificada, esta abordagem permite resultados aceitáveis, principalmente quando se conhece bem o tipo de textos a ser analisado. No entanto, é necessário identificar os padrões para cada uma das entidades, o que é inviável de fazer na totalidade, já que a linguagem permite várias maneiras de nos referirmos à mesma coisa. Outra desvantagem é o facto de os padrões encontrados serem dependentes do domínio, já que diferentes vocabulários são usados em diferentes tipos de texto.

Baseadas em dicionários

Estas abordagens dependem de um dicionário previamente disponível. Este dicionário é uma coleção de entidades que serve para fazer corresponder o texto encontrado com as entradas do dicionário.
Um exemplo desta abordagem é o sistema SIEMÊS, que utiliza um dicionário de entidades público e em língua portuguesa, o REPENTINO. Depois de encontradas as possíveis entidades, este sistema utiliza regras de semelhança entre a entidade encontrada e o dicionário para identificar as possíveis classes. Assim, o número de regras

é reduzido, sendo elas a semelhança total ou a semelhança parcial, no início, no final ou a meio da palavra.
Esta abordagem prima pela simplicidade, mas obriga a que uma entidade esteja presente no dicionário para ser reconhecida. No entanto, até um dicionário de grandes dimensões só contém uma porção das possíveis entidades. O tamanho do dicionário também pode influenciar a performance do REM.

2.2.2. Técnicas de aprendizagem computacional

Ao invés de se basear em regras escritas à mão, estas técnicas baseiam-se em modelos de aprendizagem computacional que aprendem com os dados que lhes são passados. Tipicamente estes modelos têm uma precisão superior às abordagens baseadas em regras e são mais fáceis de manter, já que podem aprender novas classificações se treinados com esses novos dados. Para treinar estes modelos é necessário um corpus linguístico, isto é, um grande conjunto de textos estruturados e anotados com os valores corretos a ser aprendidos no qual os algoritmos de aprendizagem computacional treinam as suas hipóteses para chegar à melhor precisão possível.
Há vários modelos passíveis de ser aplicados no REM, sendo os mais comuns modelos probabilísticos como Modelos Ocultos de Markov, Modelos Markov de Entropia Máxima e Campos Aleatórios Condicionais. Estes e outros modelos são apresentados de seguida.

Modelos Ocultos de Markov (MOM)

Um MOM é um modelo de Markov estatístico em que o estado não é diretamente visível. Em REM, os estados são, por norma, o nome da classe de uma entidade, com um estado extra que define se a palavra atual é uma entidade ou não. Cada transição de estado é dependente apenas no estado atual e representa a probabilidade da próxima palavra ser de uma determinada classe.
Como (Ponomareva et al., 2007) explicou, considerando $o = \{o_1, o_2, \dots, o_n\}$ uma sequência de palavras de um texto com dimensão *n.* Sendo *S* o conjunto de estados de uma máquina de estados, cada um associado a uma classe (categoria da entidade) e $s = \{s_1, s_2, \dots, s_n\}$ a sequência de estados que correspondem às categorias das palavras em *o*. MOM define a probabilidade conjunta de um estado, dada uma sequência de entrada como:

$$\mathrm{P}(s, o) = \prod_{\mathrm{i}=1}^{\mathrm{n}} P(o_i|s_i)P(s_i|s_{i-1})$$

Assim, de forma a treinar os MOM as seguintes probabilidades devem ser definidas:

(1) Probabilidades iniciais $P_0(s_i) = P(s_i|s_0)$ para começar com o estado i
(2) Probabilidades de transição $P(s_i|s_{i-1})$ para transitar do estado s_{i-1} para s_i
(3) Probabilidades de observação $P(o_i|s_i)$ do aparecimento da palavra o_i, condicionado pelo estado s_i

Modelos Markov de Entropia Máxima (MMEM)

Os MMEM funcionam de maneira semelhantes aos MOM, com a diferença que nos MMEM assumimos que os valores a ser aprendidos não são independentes entre si, podendo existir correlação. Enquanto que os MOM são modelos generativos, MMEM são modelos discriminativos. Ou seja, os MOM aprendem a probabilidade conjunta de distribuição P(*s,o*) enquanto que os MMEM aprendem a probabilidade condicional da distribuição de P(s/o).

A probabilidade de uma sequência de estados dada uma observação é então:

$$P(s|o) = \prod_{i=1}^{n} P(s_i|s_{i-1}, o_i)$$

Campos Aleatórios Condicionais (CAC)

Os modelos CAC funcionam de forma semelhante aos MOM, mas sem estarem limitados às características locais. Desta forma, os CAC são capazes de lidar com um número de características muito superior. Outra diferença é que enquanto os MOM devem satisfazer certas restrições, nos CAC não existem restrições. De acordo com (Lafferty et al., 2001), assumindo $o = \{o_1, o_2, \dots, o_n\}$ como uma sequência de palavras num texto de dimensão n. Sendo S o conjunto de estados finito de uma máquina de estados, cada um associado a uma classe (categoria da entidade). Sendo $s = \{s_1, s_2, \dots, s_n\}$ a sequência de estados que correspondem às categorias das palavras em *o*. Os CAC definem a probabilidade condicional de um estado dada um sequencia de entrada como:

$$P(s|o) = \frac{1}{Z_0} \exp\left(\sum_{i=1}^{n} \sum_{j=1}^{m} \lambda_j f_j(s_{i-1}, s_i, o, i)\right)$$

Onde Z_0 é o fator de normalização para todas as sequências de estado, $f_j(s_{i-1}, s_i, o, i)$ é uma das *m* funções que descrevem um atributo e λ_j é o peso aprendido para cada função.

Support Vector Machines (SVM)

As SVM são classificadores lineares binários não probabilísticos. Estes classificadores têm como objetivo encontrar um hiperplano de separação entre duas classes que maximize a distância entre os pontos mais próximos em relação a cada uma dessas classes.

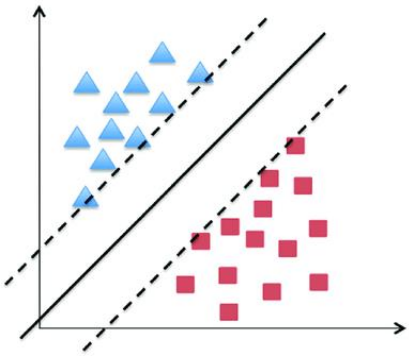

Figura 9 - Exemplo de um hiperplano que separa as duas classes

Originalmente, uma SVM apenas suporta dados linearmente separáveis, mas recorrendo a funções de *kernel* é capaz de transformar dados não linearmente separáveis no seu espaço original, para um espaço de mais dimensões onde esses dados são então linearmente separáveis. Outra limitação seria o facto das SVM apenas suportarem classificação binária (entre duas classes) mas, de forma a ser usada em tarefas não binárias como o REM, podemos adotar uma abordagem um-contra-um em que vários classificadores são criados e cada um lida com um par de classes. No fim, através dos resultados de cada classificador é escolhida a classe que melhor se adapta à entidade em questão.

2.2.3. Técnicas de ontologia

O conceito de extração de informação baseada em ontologias é relativamente recente. Para ajudar perceber como funciona o REM nestas abordagens é útil olharmos primeiro para o conceito de ontologia. Uma ontologia é um modelo de dados que representa um conjunto de conceitos dentro de um domínio e os relacionamentos entre eles. Podemos pensar numa ontologia como um dicionário (tal como nas técnicas baseadas em dicionários) mas cujos vocábulos (nesta abordagem nomeados indivíduos) têm atributos, pertencem a uma classe e estão relacionados entre si. Este tipo de modelos de dados pode ser retirados de bases de dados relacionais que por vezes já contêm esta informação.

Esta abordagem é útil em domínios mais específicos quando as classes mais comuns e gerais de classificação não chegam para detetar corretamente as entidades. Por exemplo, (Tan & Lambrix, 2009) utilizou esta abordagem em textos biomédicos em que se pretende identificar corretamente medicamentos. Neste domínio é bastante mais benéfico utilizar uma abordagem baseada em ontologia com modelos de dados que incluem o medicamento com o seu tipo, ingredientes, fabricante, sintomas, entre outros, já que o domínio é bastante específico e as classes genéricas como pessoa e organização, etc., não são suficientes.

Capítulo 3
Estado da arte

Neste capítulo são apresentadas abordagens já implementadas que acrescentam valor à criação do protótipo proveniente deste estágio. Estas abordagens dividem-se em dois tipos: ferramentas que permitem o reconhecimento de entidades mencionadas, já que esta vai ser a base da interpretação automática de bases de dados, e produtos no mercado que realizam a tarefa de interpretar automaticamente bases de dados.

3.1 Ferramentas de REM

Nesta secção são apresentadas as várias ferramentas utilizadas atualmente para solucionar tarefas de reconhecimento de entidades mencionadas e comparadas entre si. Os critérios que levaram à escolha destas ferramentas são:

1. A ferramenta deve ser gratuita.
2. A ferramenta deve suportar várias linguagens e vários domínios.
3. A ferramenta deve permitir treinar modelos personalizados com categorias de entidades personalizadas.

Com isto em mente, foram escolhidas 4 ferramentas: NLTK, OpenNLP, spaCy e Stanford CoreNLP. Estas ferramentas têm uso gratuito, sob as licenças descritas na Tabela 1. São baseadas em Python ou Java e, apesar de nenhuma suportar a língua portuguesa por definição, é possível treiná-las para que suportem outras linguagens.
Exceto o spaCy que usa o seu próprio algoritmo, estas ferramentas recorrem a algoritmos comuns e descritos no capítulo anterior como modelos markov de entropia máxima e campos aleatórios condicionais.

Tabela 1. Tabela comparativa das ferramentas de REM

FERRAMENTA	LINGUAGEM	CLASSIFICADOR	LINGUAS PRÉ-DEFINIDAS	LICENÇA
NLTK	Python	*Naive Bayes*, Árvores de decisão, MMEM	Inglês	Licença Apache
OpenNLP	Java	MMEM	Espanhol, Inglês, Holandês	Licença Apache
spaCy	Python	Modelo Linear de Thinc	Alemão, Inglês	Licença MIT
Stanford CoreNLP	Java	CAC	Alemão, Árabe, Espanhol, Francês, Inglês, Mandarim	Licença Pública Geral (GNU)

De notar que, apesar da ferramenta Stanford CoreNLP satisfazer os critérios que levaram à sua inclusão nesta análise, depois de uma inspeção mais aprofundada à sua licença com a equipa da CSW esta deixou de ser uma opção viável. A Licença Pública Geral (GNU) apesar de garantir o direito à utilização, modificação e distribuição do software, obriga

a que este se mantenha “livre” disponibilizando o código fonte e impedindo transformar um produto sob esta licença num produto proprietário.[2]

3.2 Produtos atualmente no mercado

Nesta secção são apresentados produtos presentes no mercado que realizam a tarefa de interpretar automaticamente conteúdo de bases de dados. Apesar destes produtos terem propósitos diferentes ao que se pretende neste estágio e não ser possível visualizar o seu código fonte ou tecnologias usadas, a sua análise mostra-nos o que é possível implementar e serve de inspiração para o design e funcionalidades.

SQL Server

A plataforma SQL Server é um Sistema de Gestão de Base de Dados (SGBD) relacionais desenvolvido pela Microsoft. A sua principal função é armazenar e recuperar dados solicitados por outras aplicações, sendo um dos principais SGBD no mercado, no qual existe desde 1989. A partir da versão 2019 foi adicionada uma ferramenta nomeada “Deteção e Classificação de Dados” que permite detetar, classificar, etiquetar e comunicar dados confidenciais nas bases de dados. O principal objetivo desta ferramenta é garantir a segurança da base de dados, ajudando a cumprir com as normas de privacidade, a controlar o acesso a dados sensíveis e a alertar para o acesso anómalo a esses dados.

Para tal, o SQL Server classifica os dados como empresariais, financeiros, de saúde ou pessoais. Consoante o tipo de dados encontrado é atribuída uma etiqueta de sensibilidade que divide os dados em públicos, confidenciais, muito confidenciais ou genéricos. A imagem seguinte exemplifica o funcionamento desta ferramenta.

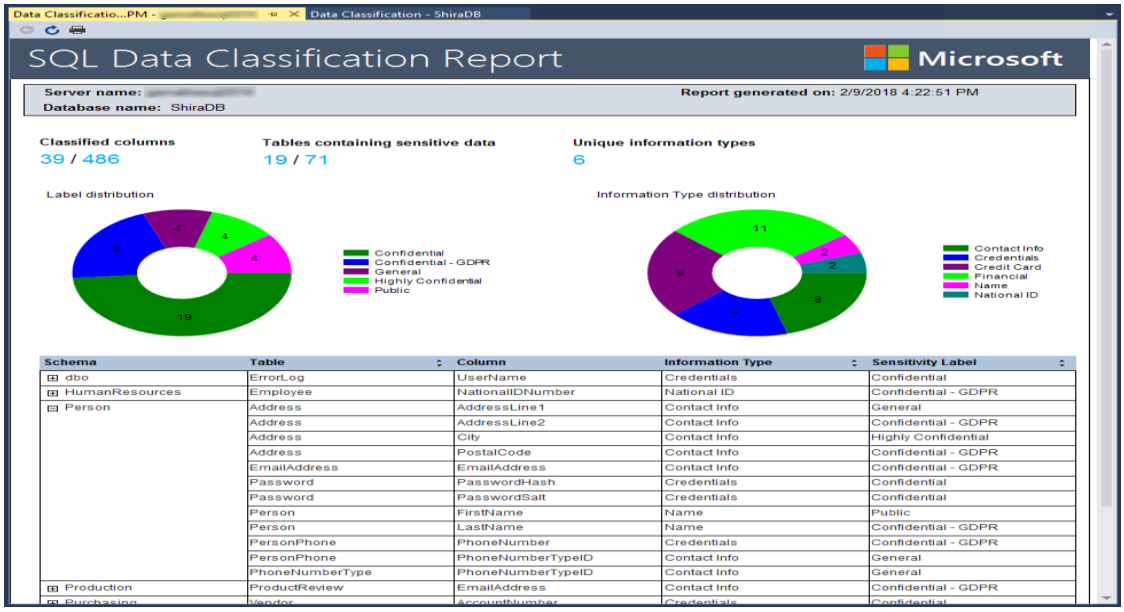

Figura 10 - Interface da ferramenta de deteção e classificação de dados do SQL Server[3]

[2] Fonte: Licenses - GNU Project - Free Software Foundation
[3] Fonte: SQL Data Discovery & Classification - SQL Server | Microsoft Docs

Os relatórios gerados pela ferramenta são apresentados como recomendações ao utilizador que pode aceitar as classificações geradas ou alterá-las.

ApexSQL Mask

O software ApexSQL é um programa que permite aplicar mecanismos de integração contínua e entrega contínua a bases de dados, bem como realizar tarefas críticas na administração desses dados. No entanto, a ferramenta mais relevante para analisar tem o nome de "ApexSQL Mask". Esta aplicação tem a capacidade de classificar os dados em categorias predefinidas como email, password, datas, número de cartão de crédito, entre outros, ou em categorias criadas pelo próprio utilizador. Esta classificação serve para aplicar máscaras às linhas ou colunas encontradas, ou seja, modificar a forma como os dados são apresentados. Por exemplo, ao encontrar números de cartões de crédito podemos querer substituí-los por sequências da letra "X" do mesmo tamanho, de forma a tornar esta informação oculta para quem não tem autorização de a visualizar.
Quanto à classificação do conteúdo, é possível verificar que as categorias criadas pelo utilizador são baseadas em regras.
A imagem seguinte demonstra a interface deste programa, na qual podemos ver a listagem das colunas de uma base de dados, bem como a categoria que lhe foi atribuída pela classificação, o tipo de dados, a máscara aplicada, entre outros. No canto direito podemos observar as alterações que a máscara escolhida causará nos valores reais.

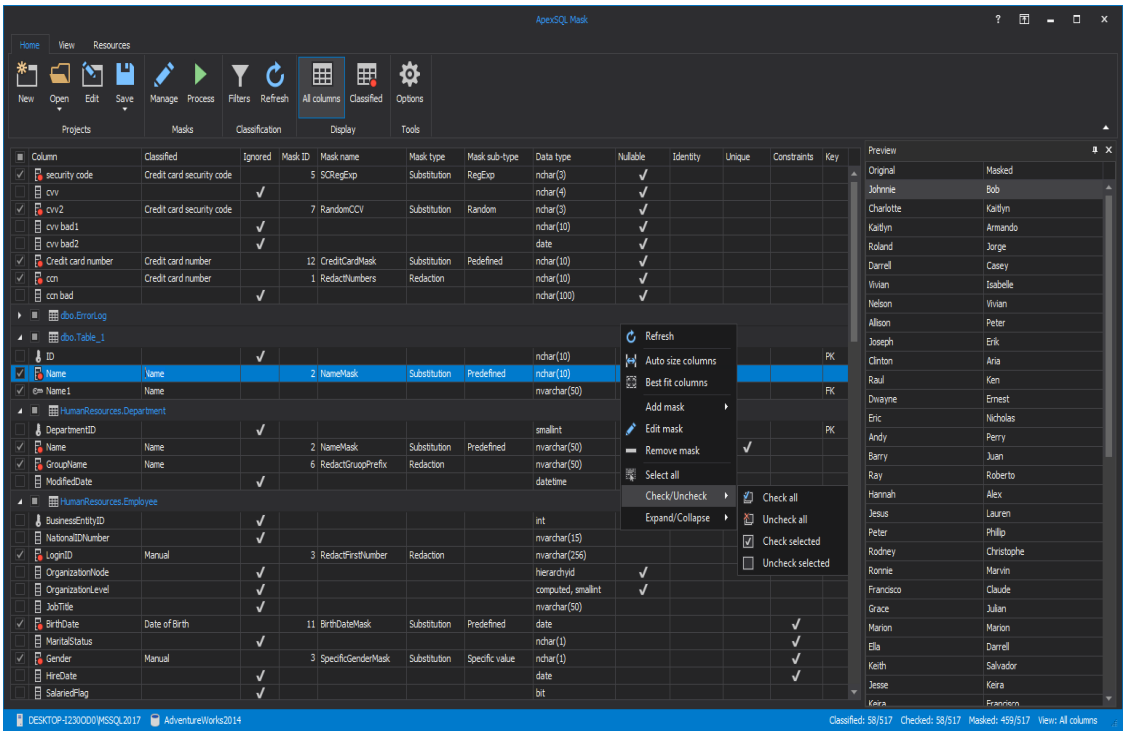

Figura 11 - Interface do programa ApexSQL Mask[4]

[4] Fonte: Data classification and masking | ApexSQL

Netwrix Data Classification

Este software, desenvolvido pela Netwrix, é o mais completo em comparação com as ferramentas já descritas. Não está apenas focado em bases de dados, suportando vários tipos de documentos. Permite classificar automaticamente dados, remover dados duplicados, criar ações que executam aquando da classificação dos dados como, por exemplo, mudar a acessibilidade a esses dados, pesquisar informações pessoais espalhadas em vários documentos, entre outros.

Na Figura 12 temos um exemplo da sua interface que nos mostra um resumo das classificações feitas. Podemos ver que a informação é classificada de acordo com a sua sensibilidade e as categorias à qual pode pertencer representam as várias regulamentações como GDPR e CCPA. É também apresentada informação acerca do tipo de ficheiros analisados e o local onde estão alojados.

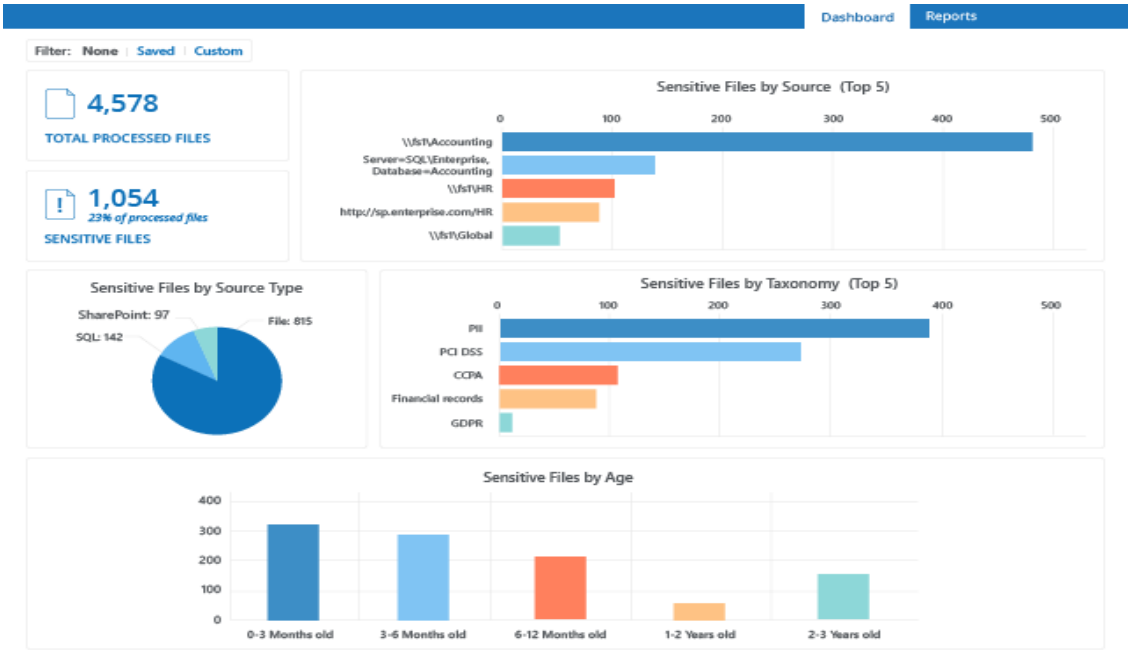

Figura 12 - Interface do programa Netwrix Data Classification[5]

A classificação dos dados é feita recorrendo a um sistema híbrido que utiliza técnicas baseadas em regras e técnicas baseadas em modelos estatísticos de aprendizagem computacional.

[5] Fonte: Data Classification Software from Netwrix

Resumo

Para facilitar a comparação entre os produtos encontrados, a Tabela 2 sintetiza as características destas ferramentas em relação às técnicas de classificação (manuais ou de aprendizagem computacional) utilizadas e em relação à forma como os dados são classificados (quanto à sua categoria ou quanto à sua confidencialidade).

Tabela 2. Tabela comparativa dos produtos disponíveis no mercado.

FERRAMENTA	TÉCNICAS DE CLASSIFICAÇÃO		CLASSIFICAÇÃO DOS DADOS	
	Manuais	Aprendizagem Computacional	Categoria	Sensibilidade
MICROSOFT SQLSERVER	?	?	✔	✔
APEXSQL MASK	✔	?	✔	✔
NETWRIX DATA CLASSIFICATION	✔	✔	✘	✔

Atualmente, qualquer organização que guarde dados pessoais, privados ou sensíveis tem a obrigação legal de provar que o faz de forma segura. Isto leva a que as ferramentas de classificação encontradas sejam focadas na segurança, para garantir que dados confidenciais não são acessíveis por entidades não autorizadas. Apesar do estágio e a plataforma ANSA terem propósitos diferentes, a análise destas abordagens contribuiu para enriquecer o conhecimento daquilo que é possível fazer e de que forma. Os sistemas de REM baseados em regras, quando aplicados a bases de dados, podem trazer bons resultados. Uma abordagem híbrida que recorre a técnicas baseadas em regras e técnicas baseadas em aprendizagem computacional, como aquela utilizada pela Netwrix, é uma metodologia bastante pertinente a este trabalho e deve ser testada.

Capítulo 4
Plataforma ANSA

Apesar do trabalho desenvolvido durante este estágio não integrar de forma direta o ANSA, faz sentido falarmos um pouco desta plataforma e percebermos o seu funcionamento.

Vivemos na era da "*Big Data*" em que a quantidade de dados que temos disponíveis é cada vez maior. Simultaneamente, a necessidade de analisar esses dados e obter respostas rápidas também é cada vez maior. Com isto em mente, a CSW criou uma plataforma web nomeada ANSA que pretende resolver estes problemas. No ANSA é possível fazer pesquisas sobre bases de dados inteiras em linguagem natural. Um pouco como acontece no motor de busca Google, aqui o utilizador pode fazer perguntas sem a necessidade de aprender linguagens formais, como SQL, e obter as respostas em tempo real. A imagem seguinte apresenta a resposta obtida quando se pedem os "5 melhores clientes em França, em 2017" a uma base de dados de vendas.

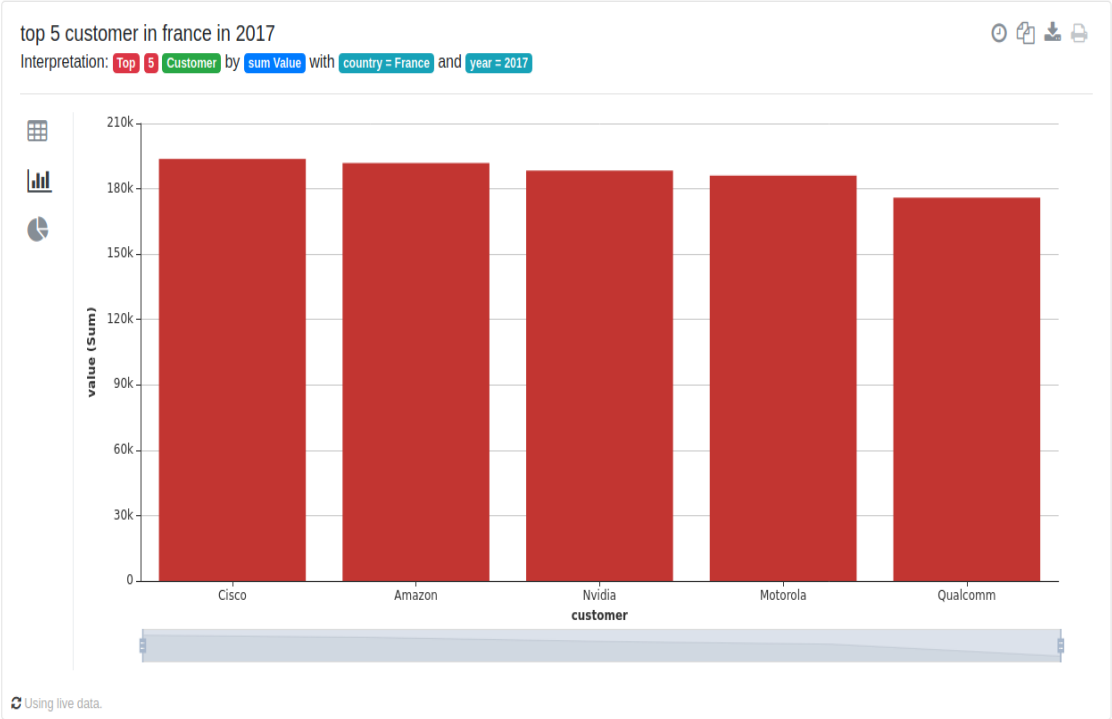

Figura 13 - Exemplo de um formato de resposta dada pelo ANSA

Não só o ANSA permite ao cliente obter a informação que pretende, como também a procura apresentar da forma mais legível possível, seja ela na forma de texto simples ou até criando gráficos automaticamente, desde gráficos de barras a gráficos circulares. A Figura 14 mostra-nos uma apresentação diferente da anterior, na forma de *pie-chart*.

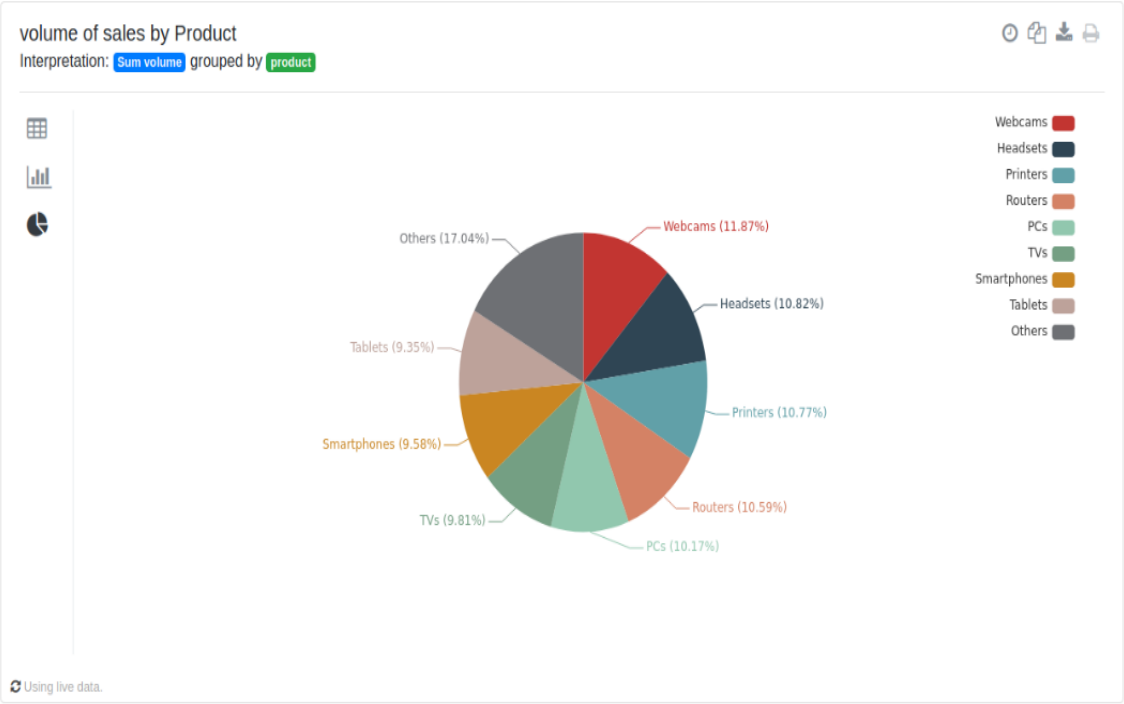

Figura 14 - Resposta dada pelo ANSA no formato de *pie-chart*

Capítulo 5
Abordagem

5.1 Requisitos

Requisitos são o conjunto de funcionalidades, propriedades e restrições esperadas para um determinado software. Dividem-se em requisitos funcionais e não funcionais, descritos de seguida.

Requisitos funcionais

Como o nome indica, os requisitos funcionais referem-se às funcionalidades que devem estar presentes no protótipo final. Nesta secção serão descritos os requisitos funcionais a implementar durante o estágio.

O principal objetivo é reduzir a complexidade e tempo necessários à configuração de uma nova base de dados, automatizando o processo. Para tal, é proposta a criação de um módulo com a capacidade de analisar e interpretar dados de uma fonte específica para proceder à sua tradução em meta dados de forma a serem usados pela plataforma ANSA (RF-1).

Para auxiliar esta configuração automática deve existir um *back office* (RF-2) constituído por uma plataforma web em que é possível criar e guardar essas configurações para que seja possível alterá-las manualmente, compará-las entre si e customizar definições de aspeto visual e de idioma.

Pretende-se ainda a criação de um painel de controlo (RF-3) com relatórios, registos do sistema e outras informações estatísticas que permitam ao administrador perceber o comportamento dos utilizadores na plataforma.

Os requisitos de alto nível acima descritos estão representados na Tabela seguinte, na qual lhes é atribuída uma prioridade. Esta prioridade segue uma escala com 3 intervalos em que o número *1* representa uma prioridade crítica, o número *2* representa uma prioridade moderada e o número *3* representa uma prioridade baixa, indicando que o requisito acrescenta valor ao protótipo final, mas que não é de caráter obrigatório.

Tabela 3. Requisitos funcionais de alto nível

ID	NOME	PRIORIDADE	DESCRIÇÃO
RF-1	Extração e interpretação automática de informação	1	Deve ser possível analisar e interpretar informação de bases de dados de forma a ser usada na plataforma ANSA.
RF-2	*Back Office*	1	Através de uma plataforma web, deve ser possível guardar e gerir as configurações automáticas descritas em RF-1.
RF-3	Painel de controlo	3	Devem ser apresentadas informações estatísticas que permitam ao administrador perceber o comportamento dos utilizadores na plataforma.

A extração e interpretação automática de informação corresponde à componente exploratória do estágio. Os requisitos para esta funcionalidade são então de alto nível de forma a permitir uma abordagem exploratória em que várias metodologias são testadas. Trata-se de um problema de classificação a ser resolvido por três tipos de sistemas: sistemas manuais baseados em regras ou dicionários, sistemas baseados em modelos e sistemas híbridos juntando estas duas abordagens (RF-1.3). Nestes sistemas, a classificação das bases de dados deve identificar a categoria de dados em cada coluna (RF-1.1), o tipo de variável (RF-1.2) e sugerir sinónimos para essa coluna (RF-1.4).

Tabela 4. Requisitos Funcionais de RF-1: Extração e interpretação automática de informação.

ID	NOME	PRIORIDADE	DESCRIÇÃO
RF-1.1	Classificar conteúdo dos dados	1	Associar os dados a categorias como “localização”, “financeiro”, entre outros.
RF-1.2	Classificar tipo dos dados	1	Classificar os dados consoante o seu tipo como sendo numéricos, nominais ou datas.
RF-1.3	Testar múltiplas abordagens aos RF-1.1 e RF-1.2.	1	Utilizar classificadores baseados em regras, em modelos de aprendizagem computacional e abordagens híbridas.
RF-1.4	Sugerir sinónimos para os nomes das colunas	2	Considerando o conteúdo de uma coluna numa base de dados, sugerir os sinónimos mais comuns que a descrevem.

A plataforma web deve esta protegida por um sistema de autenticação (RF-2.1).
A configuração automática sugerida pode não identificar na totalidade as características da base de dados. Como tal, o *Back Office* deve permitir ao utilizador visualizar e alterar manualmente a configuração proposta (RF-2.2 e RF-2.3). Deve ainda ser possível ao configurador escolher duas configurações para serem comparadas e apresentadas as diferenças entre si (RF-2.4).
Para melhorar a interpretação de futuras bases de dados deve ser possível receber feedback dos utilizadores relativo a uma configuração escolhida (RF-2.5).
As alterações de configuração estendem-se ainda ao aspeto visual e à linguagem da plataforma ANSA (RF-2.6).

Tabela 5. Requisitos Funcionais de RF-2: *Back Office*.

ID	NOME	PRIORIDADE	DESCRIÇÃO
RF-2.1	Autenticação	1	Para aceder às funcionalidades do *Back Office* é necessário primeiro fazer *login*.
RF-2.2	Capacidade de visualizar configurações	1	Permite ao utilizador visualizar as configurações guardadas.
RF-2.3	Capacidade de alterar manualmente as configurações	1	O utilizador deve poder alterar manualmente e com mais detalhe as configurações geradas automaticamente.
RF-2.4	Comparações entre configurações	2	Deve ser possível comparar as propriedades de duas configurações à escolha do utilizador.
RF-2.5	Feedback à interpretação	2	Deve ser possível atribuir feedback às configurações de forma a melhorar interpretações futuras.
RF-2.6	Menu de alterações às configurações de aspeto visual e linguagem	3	Através do *Back Office* deve ser possível alterar definições de caráter visual ou linguístico do ANSA.

O painel de controlo (RF-3) tira proveito de informação proveniente da plataforma ANSA como registos de erros, pesquisas realizadas, resultados obtidos e informações provenientes da desambiguação de palavras por parte do utilizador (RF-3.1).
Para além das estatísticas retiradas de dados pré-existentes, devem ser criadas estatísticas relativas à base de dados como tamanho da tabela, frequência de utilização de atributos, taxa de acerto entre o termo procurado e o termo presente na base de dados, entre outros, com o objetivo de identificar possíveis melhorias na configuração (RF-3.2).

Tabela 6. Requisitos Funcionais de RF-3: Painel de controlo.

ID	NOME	PRIORIDADE	DESCRIÇÃO
RF-3.1	Criação de estatísticas a partir de registos existentes	2	Reunir e interpretar os dados existentes na plataforma ANSA para criação de estatísticas.
RF-3.2	Criação de estatísticas a partir da captação de novos registos da base de dados	2	Interpretar registos provenientes da utilização da base de dados para permitir identificar eventuais melhorias na configuração.

Requisitos não funcionais

Os requisitos não funcionais correspondem aos aspetos de segurança, desempenho e prevenção de falhas, que normalmente não são visíveis durante o uso do software. Nesta secção são apresentados os requisitos não funcionais que devem estar presentes no protótipo final.

O primeiro requisito não funcional (RNF-1) remete para a necessidade de compatibilidade dos módulos implementados com Java e Kotlin, já que a atual implementação da plataforma ANSA é baseada nestas linguagens. Na eventual necessidade de recorrer a outra linguagem de programação, já que as bibliotecas necessárias ao PLN estão dependentes da linguagem, as funcionalidades implementadas devem ser acessíveis por Java e Kotlin no protótipo final.

O segundo requisito não funcional (RNF-2) indica que a ferramenta *Back Office* será baseada em Web, tal como acontece na plataforma que apoia, o ANSA.

O terceiro requisito não funcional (RNF-3) define que o *frontend* e *backend* do protótipo final são separados da implementação atual do ANSA.

A plataforma Web referida deve ser compatível com os principais navegadores como *Chrome, Firefox, Safari* e *Edge* (RNF-4).

O aspeto visual, como a disposição dos componentes e palete de cores deve estar em linha com o adotado pelo protótipo atual do ANSA (RNF-5).

O sexto requisito não funcional (RNF-6) diz respeito à configuração automática da base de dados obrigando esta a suportar todos os tipos de dados existentes no ANSA.

O sétimo requisito não funcional (RNF-7) define que a configuração automática deve suportar e ser testada com os 8 conjuntos de dados atualmente disponíveis na plataforma ANSA para teste.

O último requisito não funcional (RNF-8) leva à adição de mais 2 bases de dados, além das referidas anteriormente.

Tabela 7. Requisitos não funcionais

ID	NOME	DESCRIÇÃO
RNF-1	Compatibilidade com as linguagens Java e Kotlin	Apesar da possibilidade de recorrer a outras linguagens de programação, os resultados obtidos devem ser integrados com Java e Kotlin.
RNF-2	Ferramenta *Back Office* deve ser baseada em *web*	Analogamente ao ANSA, os módulos implementados devem ser acessíveis através de uma plataforma web.
RNF-3	Interface do utilizador e camada de suporte devem ser separadas da plataforma ANSA	O *frontend e backend* serão separados do protótipo atual do ANSA.
RNF-4	Compatibilidade com os principais navegadores	A plataforma web deve ser compatível com os navegadores mais utilizados, como *Chrome*, *Firefox*, *Safari* e *Edge.*
RNF-5	Aspeto visual semelhante aquele adotado no ANSA	O design e a palete de cores devem estar em linha com o protótipo atual do ANSA.
RNF-6	Suporte aos vários tipos de dados	A configuração automática dos dados deve suportar os vários tipos de atributos suportados pelo ANSA.
RNF-7	Suporte a todas as bases de dados de teste	A configuração automática deve suportar e ser testada em todas as bases de dados de teste.
RNF-8	Adição de 1 ou 2 conjuntos de dados relevantes	Permitir a configuração de conjuntos de dados relevantes, como por exemplo: dados Covid-19 provenientes da OMS.

Análise de riscos

Qualquer projeto de software está passível de encontrar dificuldades que ao ocorrerem podem prejudicar o produto final ou até mesmo impedir a sua conclusão e obrigar à alteração dos planos de desenvolvimento. Estas potenciais dificuldades são denominadas riscos e devem ser identificadas desde uma fase inicial para que possam ser planeadas formas de os mitigar e reduzir os seus efeitos sobre a realização do projeto. Para facilitar a sua análise, os riscos têm atributos associados como o impacto esperado, a probabilidade de acontecer e a janela temporal em que podem ocorrer.
O impacto de um risco é o efeito que esse risco tem nos critérios de sucesso do projeto. Divide-se em: *Alto* quando impede de alcançar os critérios de sucesso, *Médio* quando é possível atingir os critérios de sucesso mas com grandes dificuldades e *Baixo* quando os critérios de sucesso são atingíveis sem grandes dificuldades.
A probabilidade de um risco acontecer divide-se em: *Alta* quando o risco tem mais de 70% de probabilidade de ocorrer, *Média* quando a probabilidade de ocorrer está entre 70% e 40% e *Baixa* quando esta probabilidade é inferior a 40%.
A janela temporal de um risco corresponde ao período temporal desde a identificação do risco até ao momento em que é necessário lidar com ele. Divide-se em: *Larga* quando se estima que o risco ocorrerá num intervalo superior a 3 meses, *Média* quando esse intervalo está compreendido entre 3 meses e 1 mês e *Curta* se a janela temporal for inferior a um mês.
Os riscos identificados, bem como os seus atributos e plano de mitigação, são apresentados nas seguintes tabelas.

Tabela 8. Risco 1 - Integração e comunicação com a equipa

RISCO	Integração e comunicação com a equipa dificultada pelas medidas de saúde pública impostas face à pandemia Covid-19
IMPACTO	Alto
PROBABILIDADE	Média
JANELA TEMPORAL	Curta
PLANO DE MITIGAÇÃO	Manter o contacto pelos canais de comunicação online, participar nas reuniões remotas periódicas.

Tabela 9. Risco 2 - Familiarização com as ferramentas

RISCO	Familiarização com as ferramentas
IMPACTO	Médio
PROBABILIDADE	Alta
JANELA TEMPORAL	Curta
PLANO DE MITIGAÇÃO	Contacto com a equipa da CSW para esclarecer dúvidas sobre as ferramentas.

Tabela 10. Risco 3 - Configuração automática não permite as melhorias esperadas

RISCO	Configuração automática não permite as melhorias esperadas
IMPACTO	Alto
PROBABILIDADE	Média
JANELA TEMPORAL	Média
PLANO DE MITIGAÇÃO	Construir vários protótipos com diferentes abordagens e escolher aquela com melhores resultados. Permitir a alteração manual da configuração no *Back Office*.

Tabela 11. Risco 4 – REM não permite classificações corretas

RISCO	Tarefas de reconhecimento de entidades mencionadas não permitem a correta classificação dos dados
IMPACTO	Alto
PROBABILIDADE	Baixa
JANELA TEMPORAL	Longa
PLANO DE MITIGAÇÃO	Investigar e implementar outras tarefas de PLN e extração de informação que sejam mais capazes.

Tabela 12. Risco 5 - Tarefas não completadas dentro dos prazos

RISCO	Tarefas não completadas dentro dos prazos
IMPACTO	Alto
PROBABILIDADE	Média
JANELA TEMPORAL	Larga
PLANO DE MITIGAÇÃO	Rever mensalmente o plano de tarefas e estimar o tempo necessário. Fazer ajustes à abordagem ao estágio de forma a cumprir as metas estabelecidas.

Tabela 13. Risco 6 - Tempo para completar todos os requisitos não é suficiente

RISCO	Tempo para completar todos os requisitos não é suficiente
IMPACTO	Alto
PROBABILIDADE	Média
JANELA TEMPORAL	Larga
PLANO DE MITIGAÇÃO	Atribuir prioridades aos requisitos e completá-los por ordem de prioridade.

Tabela 14. Risco 7 - Familiarização demorada com a implementação atual do ANSA

RISCO	Familiarização demorada com a implementação atual do ANSA
IMPACTO	Médio
PROBABILIDADE	Média
JANELA TEMPORAL	Curta
PLANO DE MITIGAÇÃO	Esclarecer eventuais dúvidas com a equipa que integrou o desenvolvimento do ANSA

Tabela 15. Risco 8 - Bibliotecas disponíveis em Java não satisfazem as necessidades de implementação

RISCO	Bibliotecas disponíveis em Java não satisfazem as necessidades de implementação
IMPACTO	Médio
PROBABILIDADE	Média
JANELA TEMPORAL	Média
PLANO DE MITIGAÇÃO	Implementar numa linguagem que suporte as bibliotecas necessárias e permitir que as funcionalidades sejam acessíveis em Java ou Kotlin.

Tabela 16. Risco 9 - Falta de experiência em desenvolvimento web leva a problemas de performance ou segurança

RISCO	Falta de experiência em desenvolvimento web leva a problemas de performance ou segurança
IMPACTO	Alto
PROBABILIDADE	Média
JANELA TEMPORAL	Longa
PLANO DE MITIGAÇÃO	Replicar os mecanismos de segurança da plataforma ANSA. Enriquecer o conhecimento teórico sobre o assunto.

Capítulo 6
Metodologia e planeamento

6.1 Metodologia de trabalho

O facto de a equipa de inteligência artificial da CSW adotar a metodologia *Scrum*, aliado às condições atípicas provocadas pela pandemia Covid-19 que obrigaram ao trabalho remoto, fez desta abordagem a que melhor se adequa ao estágio. *Scrum* é uma metodologia de gestão de projetos ágil em que o trabalho se processa de forma iterativa e incremental. Isto permite que os planos de trabalho sejam alterados entre cada janela temporal definida, nomeada *Sprint*.

Os principais intervenientes nesta metodologia são o *Product Owner, Scrum Master* e a equipa de desenvolvimento. O *Product Owner* é responsável pelo produto a ser produzido, definindo os seus requisitos e priorizando as tarefas do *Product Backlog*. O *Scrum Master* é quem gere a equipa de desenvolvimento, ajudando quando necessário e garantindo que as práticas de *Scrum* são cumpridas. A equipa de desenvolvimento define a forma como vai chegar ao produto final e aquilo que é possível entregar para satisfazer os objetivos estabelecidos pelo *Product Owner*.

Os artefactos produzidos nesta metodologia são o *backlog* do produto, as *sprints*, o *backlog* das *sprints* e as entregas (incrementais) do produto. O *backlog* do produto descreve as funcionalidades que devem estar presentes no produto final. As *sprints* são intervalos temporais durante os quais se implementam as funcionalidades descritas no *backlog*. É acordada uma duração que, geralmente, vai de uma a quatro semanas. O *backlog* das *sprints* é criado nas reuniões de final de *sprint* onde podem ser feitos ajustes ao produto.

Durante o primeiro semestre, os papéis dividiram-se da seguinte forma: *Product Owner* e *Scrum Master* representados pelo Eng. Tiago Batista e a equipa de desenvolvimento representada por mim, Guilherme Pontes. Já no segundo semestre, o Eng. Rui Lopes também integrou os papéis de *Product Owner* e *Scrum Master* para melhor dividir a carga dos orientadores da CSW pelos seus orientandos.

No primeiro semestre ocorreram reuniões quinzenais com o Eng. Tiago Batista que se tornaram semanais e com a presença do Eng. Rui Lopes durante o segundo semestre. Estas reuniões corresponderam aos *sprints* onde era executado um balanço do trabalho realizado, feitos ajustes e esclarecidas dúvidas.

Ao longo do ano, aconteceram reuniões mensais com os orientadores por parte da CSW e do DEI. Os participantes foram o Professor Carlos Bento, o Eng. Tiago Batista, o Eng. Rui Lopes (2º semestre) e eu. Sendo o estágio realizado numa empresa externa ao Departamento de Engenharia Informática, estas reuniões permitiram manter as duas organizações (CSW e UC) a par do progresso realizado. Foram abordados o trabalho realizado e as dificuldades encontradas. Era também planeada a próxima *sprint*, tendo em conta o feedback dado pelos orientadores e fazendo eventuais ajustes ao planeamento de trabalho.

6.2 Planeamento

Nesta secção são detalhadas as principais tarefas desenvolvidas ao longo deste estágio. Apesar da metodologia *Scrum* não seguir um planeamento gerido por diagramas de Gantt, este tipo de diagramas foi criado para facilmente poder formular um plano, bem como analisar e adaptar as tarefas consoante o trabalho realizado em cada linha temporal. De seguida são apresentados o diagrama inicial e o diagrama final relativos a cada semestre e analisadas as suas tarefas bem como as alterações feitas.

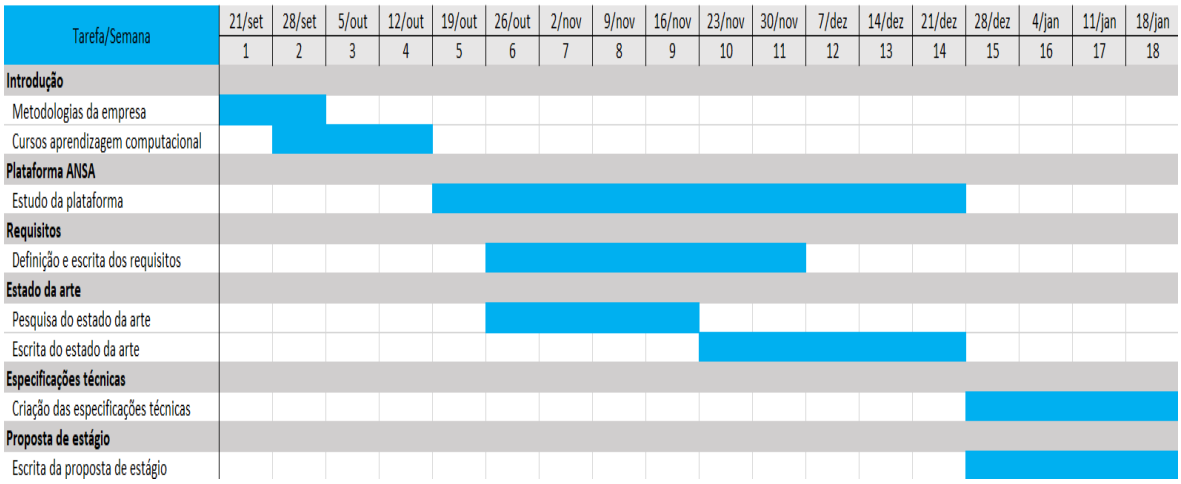

Figura 15 - Diagrama de Gantt inicial relativo ao primeiro semestre

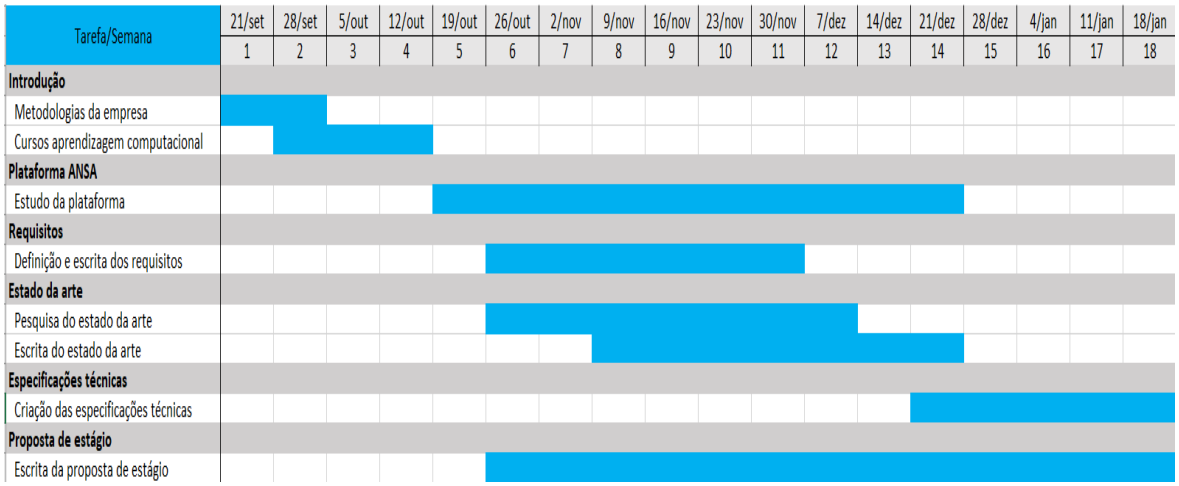

Figura 16 - Diagrama de Gantt final relativo ao primeiro semestre

As primeiras tarefas mantiveram-se inalteradas. Numa fase mais introdutória aprendi as metodologias da empresa e foi-me ensinado a utilizar as ferramentas necessárias para cumprir com essas metodologias. De seguida, para recordar alguns conceitos e acima de tudo aprender mais sobre PLN fiz os cursos online de aprendizagem computacional e de PLN lecionados por docentes da Universidade de Stanford.
Dado o âmbito desta dissertação foi necessário estudar a arquitetura e funcionamento da ferramenta ANSA. Foi também feito o levantamento e escrita dos requisitos que se manteve fidedigno ao planeamento inicial.

No entanto, como é expectável, foram necessárias algumas alterações. A pesquisa e a escrita do estado da arte passaram a ser feitas em simultâneo, isto é, depois de um início mais exploratório e de definida a estrutura considerei mais eficiente escrever ao mesmo tempo que interpretava o que lia. As especificações técnicas foram adiantadas visto o tempo definido inicialmente não ser suficiente. Já que a escrita da proposta de estágio engloba grande parte das tarefas anteriores, também aqui houve alterações ao planeamento inicial. A sua estrutura foi definida mais cedo e todas as tarefas que nela devem estar presentes foram escritas desde início com essa estrutura em mente.
Os diagramas relativos ao segundo semestre podem ser encontrados nas seguintes figuras.

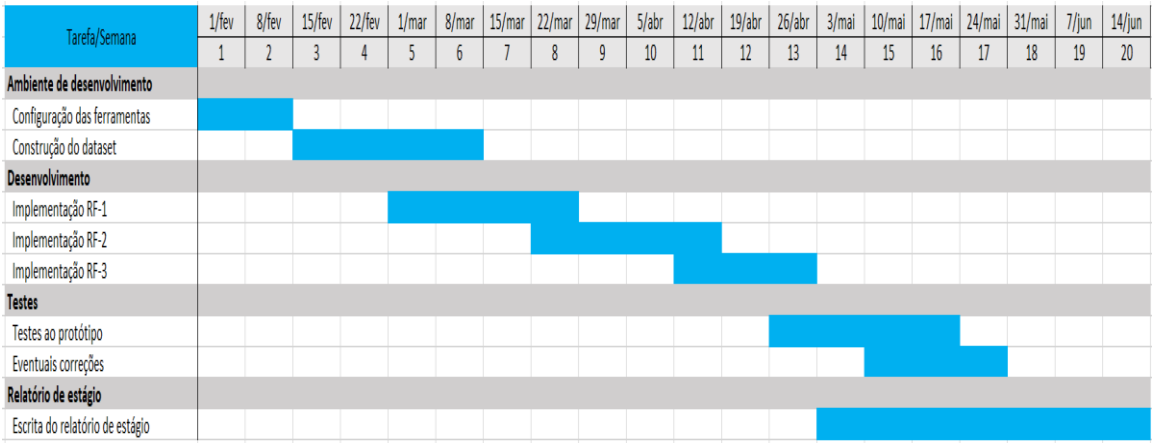

Figura 17 - Diagrama de Gantt inicial relativo ao segundo semestre

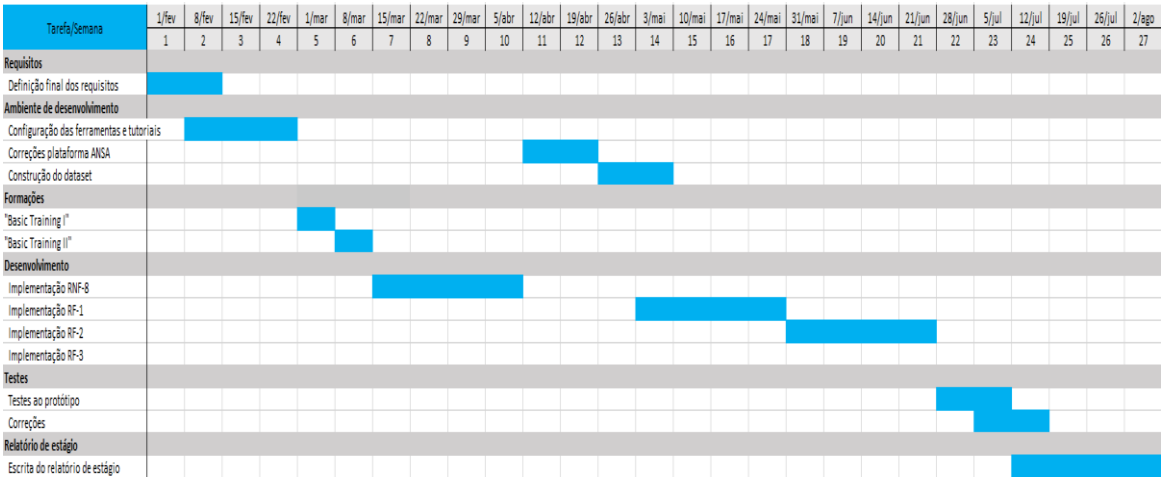

Figura 18 - Diagrama de Gantt final relativo ao segundo semestre

Comparando os diagramas, podemos ver que acresceram tarefas que não estavam planeadas inicialmente tais como as modificações feitas ao ANSA e as duas formações lecionadas pela CSW. Isto agregado ao facto de a configuração de bases de dados (RNF-8) ter demorado mais do que o esperado levou a algum replaneamento. Desse replaneamento surgiu a remoção do *frontend* que não é estritamente necessário já que o protótipo implementado vai ser usado apenas por pessoas internas à equipa de IA da

CSW. Foram também adicionados mais dois classificadores aos RF-1.1 e RF-1.2 do que os planeados inicialmente. Relativamente ao requisito RF-2 acrescentou-se também a capacidade de armazenar e gerir várias configurações e a comparação entre configurações diferentes.

Capítulo 7
Implementação

7.1 Modificações no ANSA

Antes de iniciar a implementação dos requisitos propostos para este estágio, foram necessárias algumas correções à plataforma ANSA. A versão atual do ANSA não suporta a interpretação de várias tabelas SQL, sendo necessário que estas tabelas sejam unidas numa só, denominada *LinkTable.* Este mecanismo não estava funcional e precisou de ser corrigido. Primeiramente, foram feitas correções na forma como a *LinkTable* é criada e, mais tarde, na forma como a *LinkTable* referencia os dados das tabelas originais. Por último, no *frontend* da plataforma foi adicionada a capacidade de apresentar informação no mapa dividida pelos distritos portugueses, que até agora apenas podia ser agregada por país.

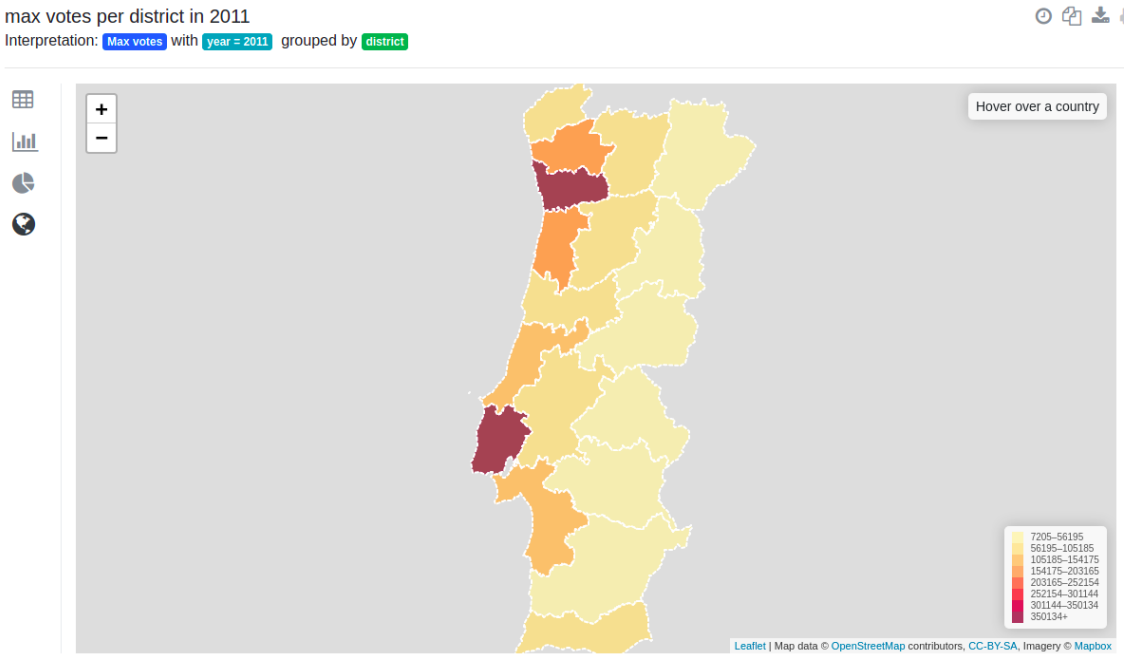

Figura 19 - Exemplo de uma pesquisa no ANSA utilizando os distritos portugueses

Realizadas estas três modificações, estavam reunidas as condições para prosseguir com a implementação dos requisitos.

7.2 Adição de bases de dados relevantes

O primeiro requisito implementado foi o RNF-8, criando manualmente e sem recurso ao *Back Office* configurações para novos conjuntos de dados relevantes. Os dados escolhidos foram relativos às Covid-19 e às eleições legislativas portuguesas.
Os dados relativos à Covid-19 são provenientes da OMS[6] e contêm informações como o número de casos, mortes, hospitalizações, testes, vacinações, rácio de transmissão, entre outros, que podem ser agregados por país e por data. Estas são algumas das pesquisas que o ANSA permite fazer com esta configuração:

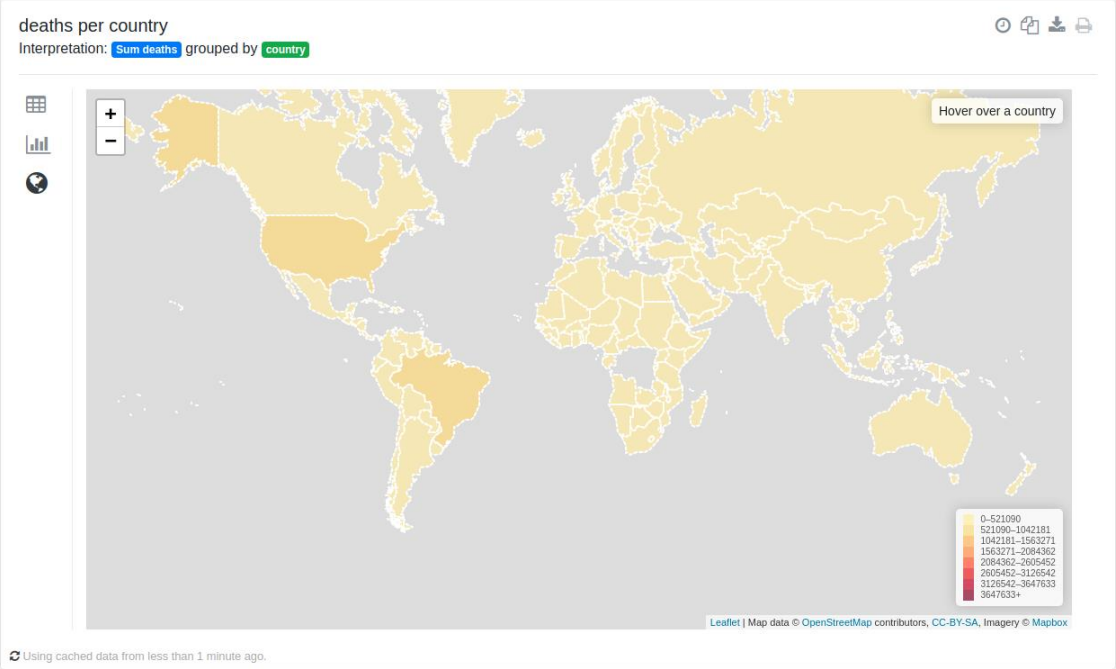

Figura 20 - Exemplo de pesquisa que mostra no mapa o número de mortos por Covid19 em cada país

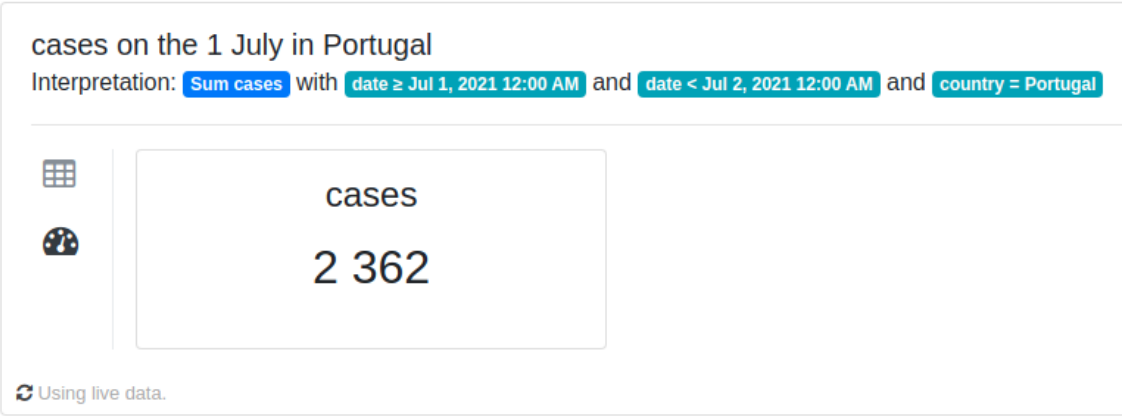

Figura 21 - Exemplo de pesquisa que mostra o número de novos casos de Covid19 em Portugal no dia 1 de julho de 2021

[6] Fonte: WHO | World Health Organization

Já a configuração das eleições legislativas portuguesas é a junção de três conjuntos de dados distintos, um deles relativo aos resultados das eleições proveniente da Central de Dados[7], outro relativo aos governos eleitos também proveniente da Central de Dados[8] e, por último, dados demográficos relevantes provenientes da PORDATA[9]. A união destes três *datasets* contém informações como os partidos eleitos em cada distrito, o número e a percentagem de votos, os governos com os seus respetivos cargos e dados relativos à população como o número de eleitores, abstenção, taxa de emprego, entre outros. Estas são algumas das pesquisas que o ANSA permite fazer com esta configuração:

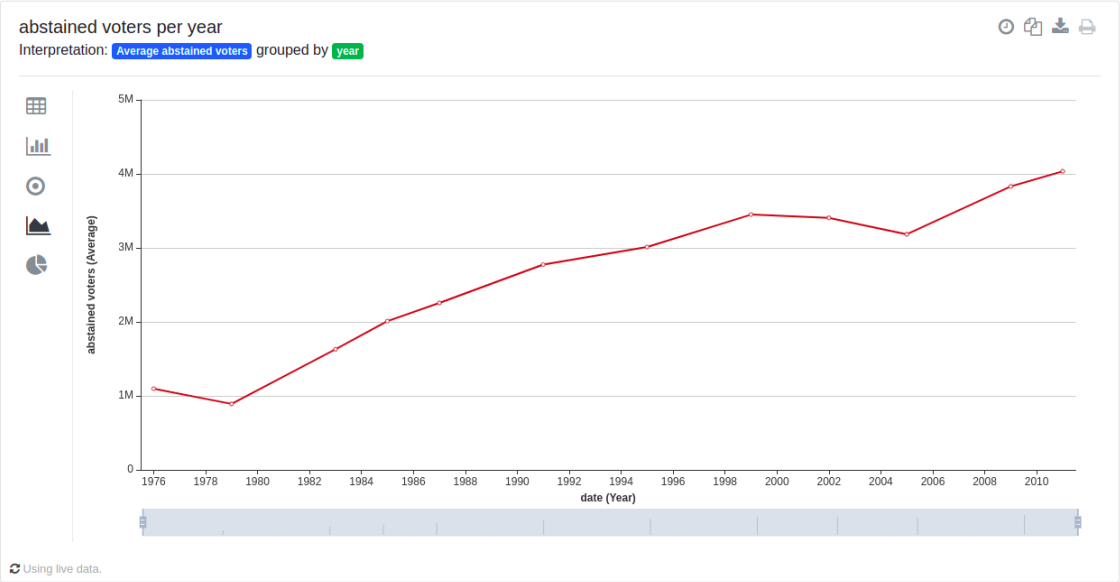

Figura 22 - Exemplo de uma pesquisa que mostra a evolução da abstenção nas eleições legislativas portuguesas desde 1976 até 2011.

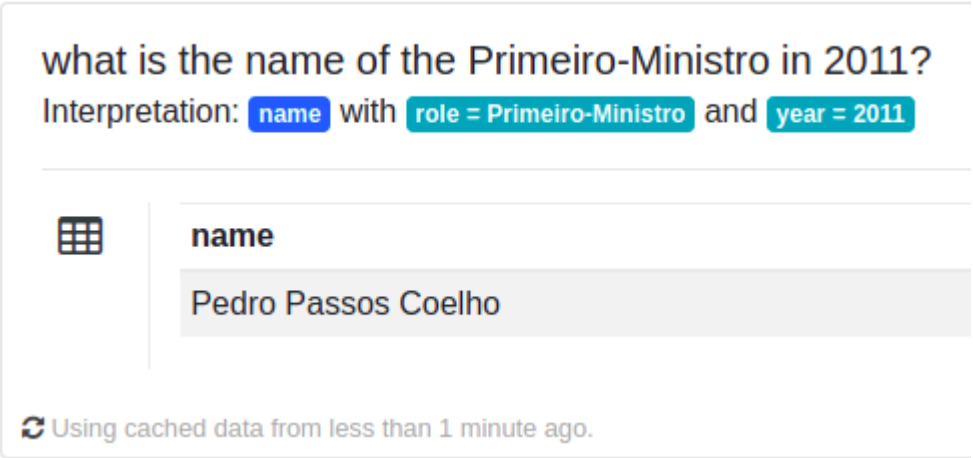

Figura 23 - Exemplo de pesquisa que apresenta o Primeiro-Ministro eleito em 2011.

[7] Fonte: Eleições legislativas — Central de Dados
[8] Fonte: Governos e Legislaturas — Central de Dados
[9] Fonte: PORDATA - Estatísticas, gráficos e indicadores de Municípios, Portugal e Europa

A escolha deste requisito como ponto de partida levou-me a aprofundar o conhecimento sobre o funcionamento do ANSA, em específico no que diz respeito à configuração de novos dados. Configurar estes conjuntos de dados manualmente, permitiu também identificar as dificuldades deste processo e perceber melhor o problema que se pretende resolver com a plataforma *Back Office.*

7.3 *Back Office*

De seguida, para automatizar o processo, foi criada a plataforma web *Back Office* (RF-2) que retira as informações necessárias à configuração da base de dados do cliente, organiza-as e as traduz numa configuração a pôr em prática na base de dados do ANSA. Para a implementação e simulação de um ambiente onde esta ferramenta será utilizada recorreu-se ao *Docker.* O *Docker* é uma plataforma para desenvolver, executar e entregar aplicações que permite separar o *software* desenvolvido do sistema onde o estamos a desenvolver. Para tal são criados "contentores", podemos pensar neles como máquinas virtuais leves e modulares que contêm o nosso projeto (ou partes dele). Desta forma, o compartilhamento de aplicações entre desenvolvedores é facilitado, evitam-se problemas de configuração e de compatibilidade entre máquinas diferentes e aumenta a velocidade com que o software é implementado no seu ambiente de produção.
Neste projeto, foram usados três contentores: um para simulação da BD do cliente, outro para simulação da plataforma ANSA e o último onde assenta a aplicação desenvolvida.

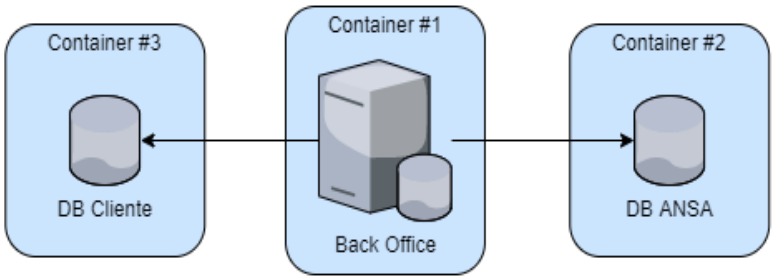

Figura 24 - Esquema da configuração *Docker* utilizada para desenvolvimento

O motor de base de dados usado nos três sistemas é o *PostgreSQL*[10]. O *Back Office* comunica com a BD do cliente para retirar as informações necessárias e construir uma configuração. Esta configuração é armazenada no *Back Office* e pode ser customizada pelo configurador. Assim que entender, a configuração é traduzida para a BD do ANSA e pode ser posta em execução assim que o ANSA for reiniciado.

7.4 Configurações

Inicialmente, o ANSA estava preparado para iniciar já com uma base de dados configurada. Foram então necessárias alterações para que inicie sem configurações já que estas passam a ser geridas e inseridas através do *Back Office* permitindo assim que o ANSA esteja a correr durante todo o processo e eliminando tempos de inatividade.
O facto de as configurações serem armazenadas no *Back Office* e só posteriormente traduzidas para o ANSA, permite a adição de algumas funcionalidades planeadas para as próximas versões do ANSA, como atributos "escondidos", ou seja, colunas da base de

[10] Fonte: PostgreSQL

dados do cliente que são necessárias ao seu funcionamento, mas que não são úteis para o utilizador final como, por exemplo, os identificadores. Nas próximas versões do ANSA estas colunas não serão apresentadas na sua interface e o *Back Office* já permite que sejam marcadas como "escondidas". Foi também acrescentada a possibilidade de dar feedback às várias configurações armazenadas e de as comparar entre si para que mais facilmente se chegue aquela que é a configuração ideal para uma determinada base de dados.
Para facilitar a compreensão da estrutura de uma configuração é apresentada a Tabela 17.

Tabela 17. Componentes de uma configuração

COMPONENTE	DESCRIÇÃO
Utilizadores	Utilizadores do ANSA com acesso à configuração, têm um nome, nome de utilizador e password encriptada com a chave utilizada no ANSA.
Fonte dos dados	Contém o endereço da base de dados do cliente para que o ANSA possa obter o seu conteúdo.
Tabelas	Define as tabelas da BD do cliente, de forma a serem interpretadas no ANSA.
Atributos	Define as colunas da BD do cliente, de forma a serem interpretadas no ANSA. É-lhes atribuído um nome a demonstrar na interface, tipo e operador de agregação.
Sugestões	Pesquisas a apresentar ao utilizador como sugestão na caixa de pesquisa, tal como acontece nos motores de busca como o *Google*.
Sinónimos	Permite associar sinónimos aos atributos previamente criados. Desta forma, quando o utilizador pesquisa pela palavra sinónima o ANSA interpreta-a como sendo a palavra original.
Substituições	Ao invés dos sinónimos que permitem pesquisas por qualquer uma das palavras, neste caso a palavra pesquisada é sempre substituída pela definida.
Anotações	Fornecem informações ao ANSA acerca do conteúdo dos atributos para melhorar a sua interpretação. Um atributo pode ser anotado como país, coordenada ou valor monetário. No caso dos países e das coordenadas permite que o resultado da pesquisa seja apresentado no mapa. A anotação de valor monetário permite associar as pesquisas relativas a dados financeiros ao atributo em anotado.
Configurações de linguagem	São customizadas opções de linguagem, como o idioma a usar pelo ANSA.
Configurações de sistema	São customizadas opções de sistema, como a quantidade de valores a manter em memória, ativar/desativar os mapas, definir o dia a ser interpretado como "hoje", personalizar as cores da interface, entre outros.

Uma configuração como a descrita em cima pode ser obtida de forma automática ou de forma manual, consoante a personalização e detalhes exigidos pelo configurador. O funcionamento destes dois métodos é apresentado de seguida.

Configuração automática

Os dados essenciais ao funcionamento de uma nova configuração são interpretados de forma automática. O objetivo é, a partir desta interpretação automática, obter uma configuração funcional que sirva como base para eventuais alterações manuais, consoante as necessidades do configurador.

O diagrama de fluxo apresentado de seguida sumariza os processos lógicos que devem decorrer de forma a configurar uma nova base de dados.

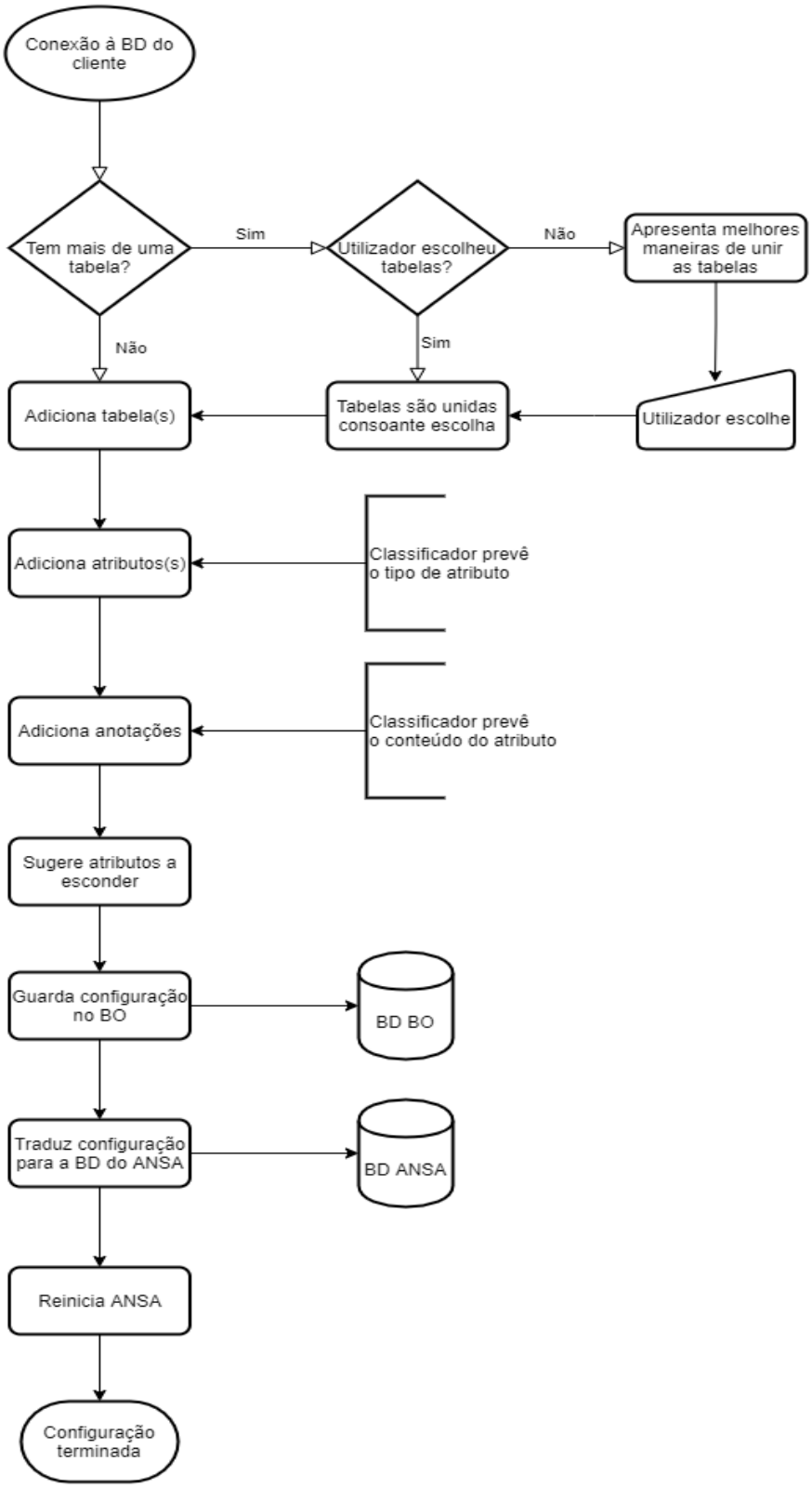

Figura 25 - Diagrama de fluxo com as operações executadas durante a configuração automática

Como referido anteriormente, o ANSA apenas suporta uma tabela SQL, portanto quando existem dados de tabelas distintas existe a necessidade as unir. A forma como as tabelas são unidas pode ser definida *à priori,* pelo utilizador, ou pela própria configuração automática que está preparada para sugerir a união que permite manter a maior quantidade de dados, se possível, agregando todas as tabelas numa só. Isto foi possível recorrendo a um algoritmo de pesquisa em largura onde é criado um grafo em que as tabelas da BD a configurar são os vértices e as chaves estrangeiras as arestas. O objetivo desta pesquisa é encontrar o caminho mais longo começando a partir de qualquer vértice, ou seja, encontrar a forma de unir o máximo número de tabelas possível numa só para que esta seja interpretada pelo ANSA. Depois das uniões definidas, ou caso exista apenas uma tabela, esta é adicionada à configuração.

Com a tabela adicionada à configuração, segue-se a adição das suas colunas. No ANSA, as colunas das tabelas SQL são nomeadas atributos e para que sejam interpretados devem referenciar a tabela e a coluna a que pertencem, ter um nome a apresentar na interface, ter um tipo e uma agregação predefinida. Estes atributos podem ainda ter sinónimos associados, ou seja, palavras com significado semelhante que o ANSA irá associar ao mesmo atributo. Estes sinónimos são sugeridos ao configurador recorrendo à biblioteca NLTK[11], com a base de dados lexical WordNet[12]. Esta combinação permite-nos obter um conjunto de sinónimos para uma determinada palavra passada como argumento. Por último, é definido o nome a apresentar na interface para representar este atributo. Para tal, recorre-se a uma função simples que formata o nome original da coluna retirando quaisquer hífens e adicionando espaços e maiúsculas.
Para que o tipo dos atributos seja interpretado de forma automática recorreu-se a classificadores, descritos na secção 7.5.
Associado aos atributos, temos ainda a noção de anotação. Uma anotação é uma forma de melhorar a interpretação do conteúdo de determinados atributos. Existem três tipos de anotações: anotações de países, de coordenadas e de moedas. Analogamente ao tipo de atributo, a interpretação automática de anotações recorre a classificadores descritos na secção 7.5.
Quando, na mesma configuração, existem vários atributos do tipo data existe então a necessidade de definir qual deles é o predefinido. Para tal, antes de concluir a configuração automática é feita uma pesquisa a todas as colunas do tipo data que escolhe como predefinida aquela que contém as datas mais recentes. Posto isto, está concluída a configuração automática.

Configuração manual

A configuração manual, para além de permitir adicionar ou alterar os parâmetros da configuração automática, permite ainda configurar funcionalidades apenas necessárias em casos específicos ou que são preferências de cada configurador. Estas funcionalidades englobam a criação de utilizadores no ANSA, adição do calendário fiscal (para configurações que têm dados fiscais), adição de sugestões de pesquisa, personalização de aspeto visual e de linguagem, entre outros, descritos no esquema seguinte. Resumindo,

[11] Fonte: Natural Language Toolkit — NLTK
[12] Fonte: WordNet | A Lexical Database for English

todas as propriedades configuráveis estão acessíveis através da configuração manual e aquelas já definidas pela configuração automática podem ser alteradas. No capítulo referente à API (secção 7.6), estão apresentadas as várias possibilidades de acesso a esta configuração.
Assim que a configuração estiver concluída, é possível implementá-la no ANSA. Para tal, o *Back Office* traduz a configuração escolhida numa configuração compatível com a versão atual do ANSA, que de seguida é guardada na sua base de dados e o ambiente é reiniciado para que as alterações entrem em vigor.

7.5 Classificação

Objetivo

Como referido na secção 7.4, durante a configuração dos atributos existe a necessidade de traduzir o tipo da coluna na base de dados num tipo de dados existente no ANSA. O ANSA suporta três categorias de dados: nominais para dados categóricos, numéricos para valores contínuos ou discretos não categóricos e datas.
Somando ao problema anterior, existe também a necessidade de classificar os dados quanto ao seu conteúdo para que sejam criadas as anotações. Aqui, interessa perceber quais são as colunas na base de dados que representam um país, um valor monetário ou uma coordenada (latitude ou longitude). Inicialmente, visto esta classificação tratar dados linguísticos, indiciava poder ser resolvida através de REM (secção 2.2). Contudo, a sua pesquisa e implementação, provaram não ser esta a solução ideal. Foi utilizado o modelo pré-treinado de REM da biblioteca spaCy[13] para fazer alguns testes preliminares que, apesar de reconhecer quando os dados são referentes a países, falha quando tentamos classificar coordenadas ou valores monetários. Isto deve-se ao facto de os modelos de REM estarem preparados para classificar dados alfabéticos presentes em frases completas e neste caso os dados são numéricos e sem contexto frásico. Concluindo, optou-se por uma abordagem tradicional de classificação, cujo procedimento está descrito nas próximas secções.
Os dois problemas de classificação referidos podem então ser abordados como um só ou de forma distinta. De forma distinta procedemos primeiro à classificação quanto ao tipo de dados (numéricos, nominais ou datas) e, de seguida, quanto ao seu conteúdo (países, dados monetários, latitude, longitude ou sem anotação). Esta abordagem leva-nos a dois classificadores, o primeiro com 3 classes e o segundo com 5.
Unindo os dois problemas num só, passamos a necessitar apenas de um classificador que distinga 7 classes (numéricos, nominais, datas, países, dados monetários, latitudes e longitudes). Nesta abordagem, assumimos que uma coluna classificada como país é do tipo nominal e do tipo numérico quando classificada como latitude, longitude ou valor monetário.
Ambas as interpretações foram testadas e foi escolhida aquela cujos testes apresentaram melhores resultados.

[13] Fonte: spaCy

Conjunto de dados

Como é habitual num problema de classificação de aprendizagem supervisionada, é necessária a criação de um *dataset*, ou seja, um conjunto de dados tabulados em que cada linha representa uma amostra e cada coluna representa as características dessa amostra. O conjunto de dados criado para solucionar este problema deve conter características que permitam ao classificador inferir alguma relação ou dependência entre essas mesmas características e a classe a que corresponde a amostra.

De forma a garantir que os resultados obtidos durante os testes dos classificadores sejam fidedignos aos alcançáveis no ambiente real, todas as características do conjunto de dados devem ser possíveis de obter durante a criação de novas configurações. Os dados e meta dados presentes neste *dataset* são prevenientes de configurações existentes disponíveis, resultando num total de 234 exemplos.

Estes dados obtidos através de configurações pré-existentes continham muito poucos exemplos de algumas classes, como é o caso da classe latitude e longitude, com apenas dois exemplos cada. Visto que se tenciona usar validação cruzada e que cada subdivisão do conjunto de dados contenha todas as classes, optou-se por acrescentar dados artificiais. Os dados artificiais procuram ser o mais fidedignos possível aos dados reais e consistem em 10 amostras da classe latitude, 10 amostras da classe longitude e 10 amostras da classe nominal. No caso da latitude e longitude trata-se de colunas com nomes indicativos de coordenadas com valores aleatórios entre -90 e 90 e -180 e 180, respetivamente. Já no caso dos nominais, o objetivo foi adicionar amostras em que os valores sejam numéricos, mas a sua interpretação deva ser nominal, isto é, dados como os séculos, grupos de idades, anos escolares, identificadores de género, entre outros que são de natureza numérica com baixo número de valores distintos mas que devem ser interpretados como nominais para que possam ser feitas agregações. Com estes novos dados, o objetivo é dar aos classificadores a capacidade de aprender a distinguir entre os valores numéricos que, no ANSA, devem ser considerados nominais e os que devem ser considerados numéricos. Com esta mudança as percentagens das classes latitude e longitude passaram de 0,8% para 4,5% e da classe nominal de 38,6% para 42,4%.

Na sua versão final o *dataset* contém 264 amostras, com as características descritas de seguida:

Tabela 18. Descrição do conjunto de dados

CARACTERISTICA	DESCRIÇÃO	EXEMPLO
Nome	Nome da coluna na BD do cliente	locations
Sigla	Sigla formada pelas primeiras 3 ou 4 letras do nome	loc
Tipo	Tipo da coluna na BD do cliente	varchar
Distintos	Número de valores distintos em toda a coluna	70
Mínimo	Valor mínimo presente nos dados da coluna, caso sejam numéricos	-90
Máximo	Valor máximo presente nos dados da coluna, caso sejam numéricos	2550
Moda	Valor que mais se repete na coluna	Portugal
Tipo ANSA	Tipo com que a coluna foi configurada no ANSA (usado como classe)	NOMINAL
Anotação	Anotação com que a coluna foi configurada no ANSA (usado como classe)	COUNTRY

Também foi testada a hipótese de retirar 200 amostras aleatórias de cada coluna a classificar, mas isto levaria à criação de características que podem ser simultaneamente alfabéticas e numéricas o que as invalida de serem interpretadas pelos classificadores.

Relativamente ao balanceamento do conjunto de dados, a abordagem de classificação única (quanto ao tipo e conteúdo dos dados simultaneamente) tem a seguinte distribuição:

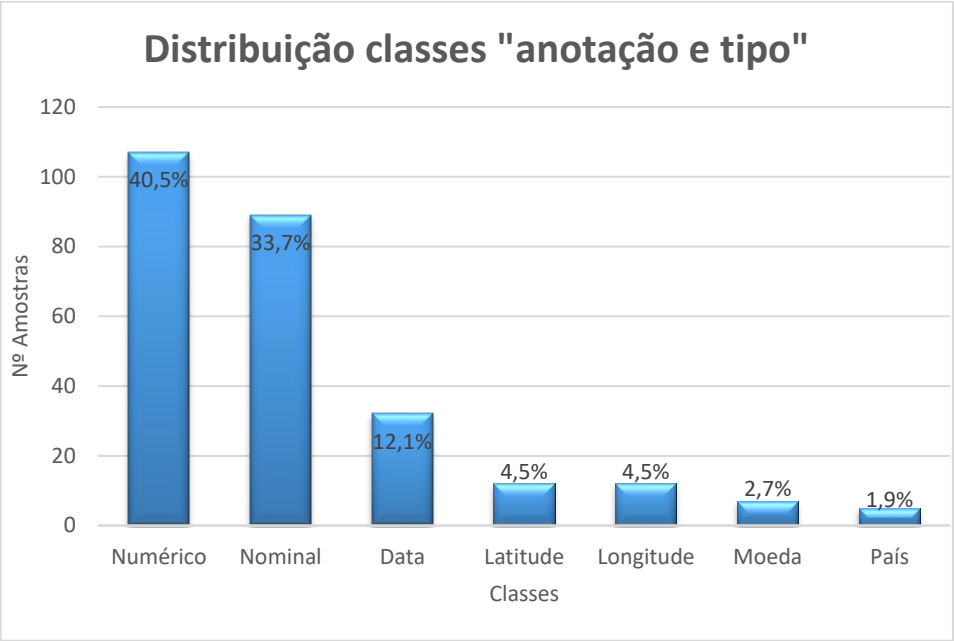

Figura 26 - Histograma com a distribuição da classe "anotações e tipo"

Na abordagem de classificação em série (primeiro quanto ao tipo dos dados e de seguida quanto ao seu conteúdo) as distribuições são as seguintes:

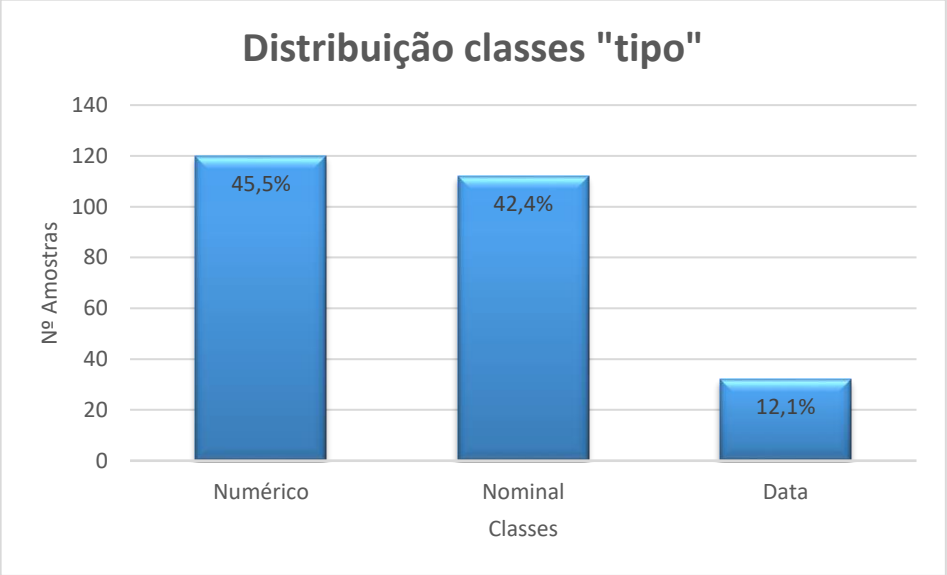

Figura 27 - Histograma com a distribuição da classe "tipo"

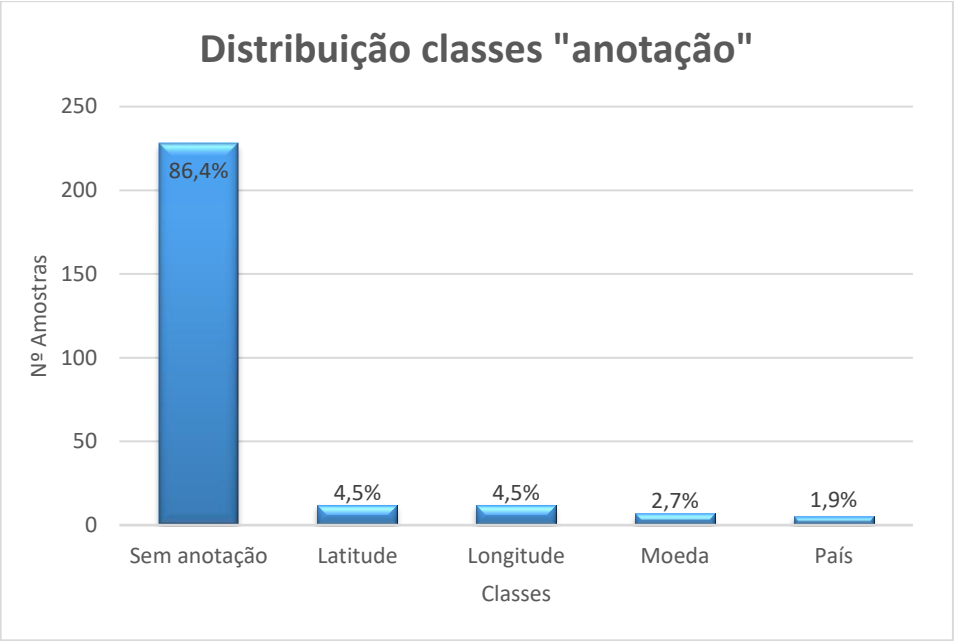

Figura 28 - Histograma com a distribuição da classe "anotações e tipo"

Atendendo a estas proporções, conclui-se que em ambas as abordagens estamos perante um *dataset* desbalanceado e devem ser tomadas medidas para evitar tendências por parte dos modelos de aprendizagem computacional.

Pré-processamento dos dados

Esta fase está compreendida entre a obtenção dos dados e a sua classificação e é de extrema importância já que afeta diretamente a capacidade de aprendizagem dos modelos. Durante esta etapa, procura-se limpar e formatar os dados de modo a que algoritmos diferentes possam ser executados sobre os mesmos dados e que estes retirem o máximo de informação possível dos mesmos. A biblioteca usada para o efeito foi o *scikit-learn*[14] com recurso à classe *Pipeline*[15]. Esta classe permite, de forma simples e legível, criar uma sequência de transformações ao *dataset* e, de seguida, classificar esses dados recorrendo a um modelo escolhido.

O primeiro passo começa por garantir que não existem valores nulos. Práticas comuns passam por eliminar as amostras com dados em falta ou pela substituição desses dados pela média dos valores dessa coluna. Porém, neste caso, a eliminação de dados levaria a uma perda significativa de informação. Tendo em consideração que as características com valores em falta são as colunas *média, mínimo* e *máximo* nos casos em que os seus conteúdos são alfanuméricos, é mais representativo substituir estes valores por zero do que pela média da coluna. Assim, os valores em falta são transformados em valores predefinidos, zero no caso das características numéricas e *"missing"* no caso das características categóricas.

Posteriormente, os valores numéricos são normalizados, isto é, cada característica é transformada de forma a seguir uma distribuição normal cuja média é zero e o desvio padrão é um.

[14] Fonte: scikit-learn: machine learning in Python
[15] Fonte: sklearn.pipeline.Pipeline — scikit-learn documentation

Visto que os classificadores apenas interpretam valores numéricos, as características alfabéticas também carecem de pré-processamento. Uma prática comum seria recorrer à codificação *one-hot* mas as suas limitações teóricas verificaram-se nos testes realizados, aumentando consideravelmente o número de características com muitos valores a 0, reduzindo a performance nos tempos de treino e na classificação. Tendo isto em consideração, para as características alfabéticas cuja quantidade de valores distintos é elevada foram utilizadas *word embeddings*, especificamente para as colunas "nome" e "sigla". Foi escolhida a biblioteca *FastText*[16] recorrendo ao seu modelo pré-treinado para *word embeddings* na língua inglesa. Desta forma, as palavras presentes no *dataset* original são transformadas em vetores de números reais com 300 dimensões em que cada dimensão procura representar o significado e o contexto da palavra. O resultado são vetores que quanto mais semelhantes forem, mais semelhantes são as palavras que representam. A principal desvantagem deste procedimento é o elevado número de características que cria, o que unido ao facto do conjunto de dados não ser de grande dimensão pode dificultar a tarefa dos modelos em conseguirem encontrar as relações necessárias para dividir as classes corretamente. Para colmatar esta limitação, são experimentadas várias técnicas de redução de dimensionalidade descritas de seguida.
Contudo, quando a quantidade de valores distintos é baixa, tal como acontece na coluna "tipo", recorreu-se a uma codificação ordinal[17] em que cada palavra distinta é traduzida num valor inteiro que a passa a representar.

- Feature engineering

Algumas experiências com *feature engineering* foram testadas. As colunas numéricas e contínuas "distincts", "avg", "min" e "max" foram agregadas de forma a tornarem-se valores discretos. Recorreu-se ao método *KBinsDiscretizer*[18] para unir em grupos os valores de cada uma destas características e assim reduzir a quantidade de valores distintos em 90%.

- Seleção e redução de características

Para além das transformações referidas, foram ainda implementadas técnicas de seleção e redução de características visando simplificar os modelos, reduzir os tempos de treino e evitar a "maldição da dimensionalidade"[19]. A seleção de características compreende apenas a escolha ou exclusão de características sem que estas sejam alteradas, ao invés da redução de características que pretende reduzir a dimensionalidade aplicando transformações aos dados originais.
Como seleção de características foi utilizado o método *SelectKBest*[20]. Este método seleciona características recorrendo a testes estatísticos, neste caso, baseando-se no valor de F do teste ANOVA entre a característica em questão e a classe. A quantidade de características a manter é escolhida pelo utilizador.
Como redução de características foram usados os métodos *Linear Discriminant Analisys*[21] (LDA) e *Principal Component Analisys*[22] (PCA). O LDA é um algoritmo supervisionado que diminui o número de características inicial do *dataset* para *c*-1, onde *c* é o número de classes. Os dados são projetados num subespaço linear que consiste nas direções que maximizam a separação entre classes.

---

[16] Fonte: fastText
[17] Fonte: sklearn.preprocessing.OrdinalEncoder — scikit-learn documentation
[18] Fonte: sklearn.preprocessing.KBinsDiscretizer — scikit-learn documentation
[19] Traduzido de *curse of dimensionality*
[20] Fonte: sklearn.feature_selection.SelectKBest — scikit-learn documentation
[21] Fonte: sklearn.discriminant_analysis.LinearDiscriminantAnalysis — scikit-learn documentation
[22] Fonte: sklearn.decomposition.PCA — scikit-learn documentation

Por outro lado, o PCA não é supervisionado e permite ao utilizador escolher a variância ou o número de características a manter. O objetivo é descartar as características menos significativas para que a perda de informação seja mínima, através de transformações ortogonais que convertem um conjunto de observações possivelmente correlacionadas num conjunto de variáveis linearmente não correlacionadas chamadas componentes principais.

Classificadores

Com o objetivo de ir ao encontro da melhor performance possível, foram implementados cinco classificadores com comportamentos distintos e consequentes adaptações distintas ao problema: baseado em regras, *K-Nearest Neighbor* (KNN), *Random Forest* (RF), *Support Vector Machines* (SVM) e *Light Gradient Boosting Machines* (LGBM).

- Baseado em regras

O classificador baseado em regras foi o primeiro a ser implementado, já que é o mais simples e a sua performance serve como base de comparação com os modelos de aprendizagem computacional. Quando o objetivo é classificar os dados quanto ao seu tipo o raciocínio é bastante simples: um dicionário faz o mapeamento do tipo da coluna na BD original para o tipo a ser considerado pelo ANSA, como demonstra o seguinte esquema:

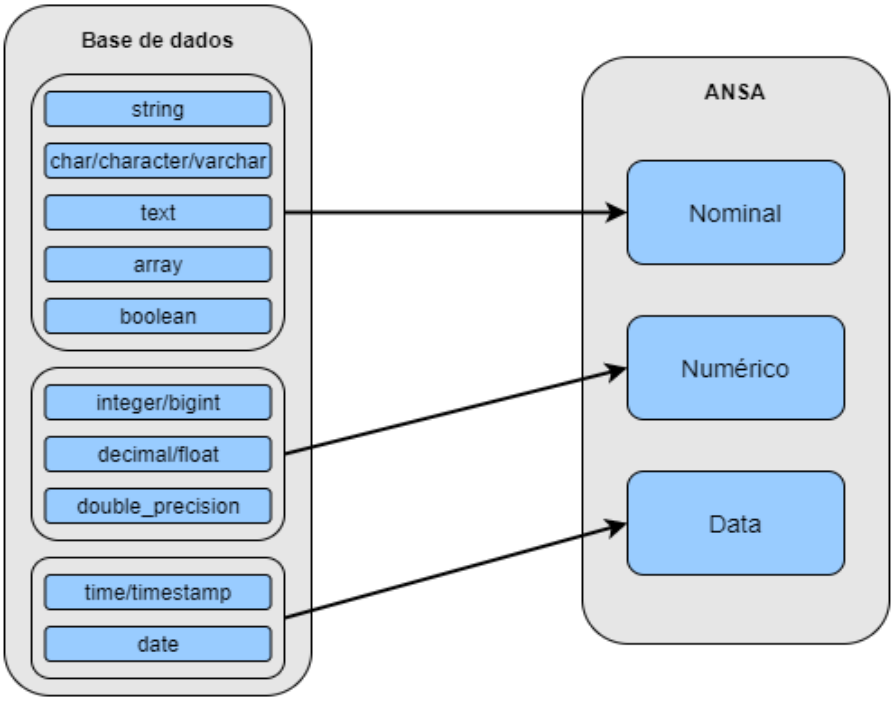

Figura 29 - Tradução dos tipos de dados na BD para os tipos de dados no ANSA

Todavia, quando o objetivo é classificar os dados quanto ao seu conteúdo é imperativo ter mais regras em consideração do que apenas o tipo dos dados. O fluxograma ilustrado na Figura 30 esquematiza os procedimentos até chegar a uma classificação. Esses procedimentos funcionam da seguinte forma: caso a coluna seja do tipo nominal podemos estar perante uma coluna de países, caso a coluna seja do tipo numérico podemos estar perante uma coluna de coordenadas ou valores monetários. Com isto em mente, o nome das colunas é, de seguida, analisado. Esse nome é comparado com uma lista de sinónimos

de países, de coordenadas ou de valores monetários e caso haja correspondência, prosseguimos com a classificação.
Por último, resta analisar o conteúdo dos dados, retirando amostras aleatórias não nulas desses dados e fazendo comparações. Para os países, foi usada a biblioteca *PyCountry*[23] que permite procurar países pelo seu nome oficial, nome comum e código ISO de 2 ou 3 carateres. Caso pelo menos 85% das amostras sejam reconhecidas como países, a coluna em questão é então classificada como uma coluna de países. Para as coordenadas, basta verificar se os valores mínimo e máximo das amostras estão compreendidos entre -90 e 90 no caso da latitude ou entre -180 e 180 no caso da longitude.

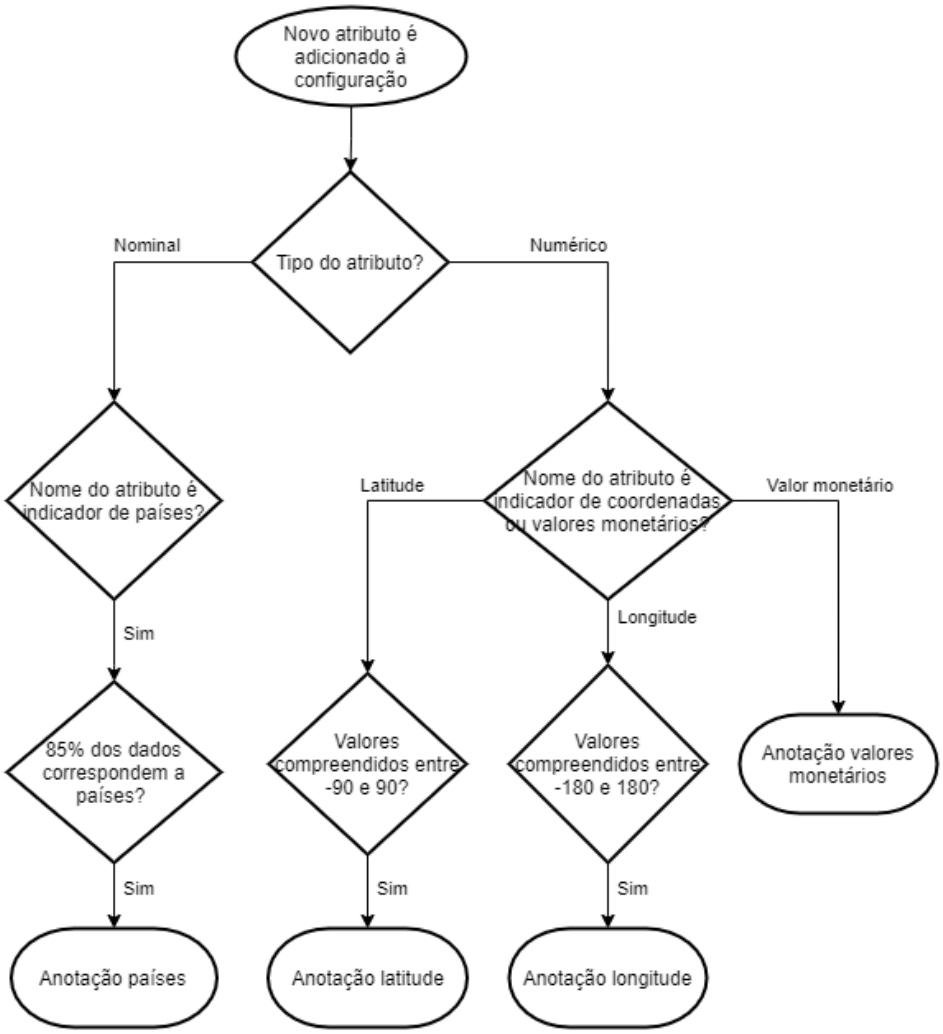

Figura 30 - Diagrama de fluxo com as regras de classificação de anotações

[23] Fonte: pycountry · PyPI

- *K-Nearest Neighbors*

O KNN é um algoritmo muito utilizado na ciência de dados, por vezes até como base de comparação com outros algoritmos dada a sua simplicidade e rapidez. Ele atua sob a premissa de que semelhança implica proximidade, ou seja, dispondo vários pontos num referencial, aqueles mais próximos são mais semelhantes entre si. Para o cálculo desta proximidade métodos como a distância Euclidiana ou a distância de Manhattan são os mais comuns. Para definir quantas distâncias devem ser tidas em consideração existe o valor de K, que também deve ser parametrizado.
Este classificador representa então um classificador baseado em distâncias e foi implementado através da biblioteca *scikit-learn.*

- *Support Vector Machines*

O principal objetivo das SVM é encontrar um hiperplano no espaço *n*-dimensional (em que *n* representa o número de características) que separe as amostras de acordo com a classe a que pertencem, ou seja, pretende-se que de cada lado do hiperplano se encontrem amostras apenas de uma classe.
A escolha das SVM deve-se à sua boa performance em *datasets* com poucos dados. Apesar de ser originalmente binário, este classificador foi implementado através da biblioteca *scikit-learn* que o transforma em multiclasse através do método *one-vs-one.*

Os dois últimos classificadores são baseados em métodos *ensemble,* isto é, combinam vários modelos base num modelo apenas, para melhorar a generalização e robustez do mesmo. Os métodos *ensemble* podem ainda ser divididos em duas famílias: os métodos *averaging* e os métodos *boosting*.
Os classificadores escolhidos, apesar de serem ambos baseados em árvores de decisão, pertencem cada um a uma família distinta.

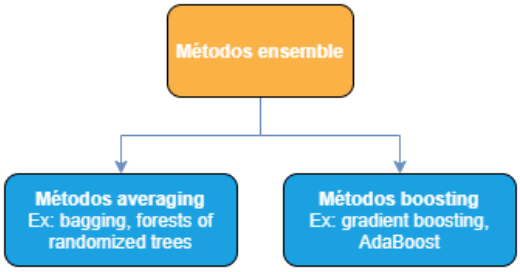

Figura 31 - Divisão dos métodos "ensemble"

- *Random Forests*
Este classificador pertence aos métodos *averaging*, isto é, tem como principal objetivo construir vários modelos base independentes (normalmente com parâmetros diferentes) para depois unir os seus resultados num só classificador. Neste caso, cada árvore de decisão da RF é treinada com dados de treino diferentes recorrendo a métodos de *bagging*. *Bagging* altera os dados de treino iniciais por substituição, mantendo as dimensões iniciais. Por exemplo, se inicialmente tivermos os dados [1, 2, 3, 4, 5], recorrendo a bagging uma árvore de decisão pode vir a ser treinada com os dados [1, 2, 2, 4, 3] em que a quantidade de dados não é alterada, mas o seu conteúdo varia por substituição.
Cada árvore pretende distinguir dois nós através da característica que mais separação produz entre eles. No caso das RF é comum que uma árvore apenas possa escolher essa

característica a partir de um subgrupo de características. Isto leva a uma maior variância entre árvores.
Os dois fatores de aleatoriedade referidos permitem obter árvores menos correlacionadas entre si e que se "protegem" umas às outras dos seus erros, isto é, os erros variam em direções diferentes evitando tendências, já que as árvores são mais diversas e baseadas em características diferentes.
Uma vantagem deste modelo é o facto de prevenir *overfitting,* caso existam árvores suficientes para o número características. Porém, um aumento no número de árvores também torna o algoritmo mais lento.
Este classificador foi implementado através da biblioteca *scikit-learn.*

- Light Gradient Boosting Machines
Este classificador pertence aos métodos *boosting*. Nestes métodos, são construídos vários modelos base, mas de forma sequencial em que cada modelo construído tenta reduzir a tendência do modelo combinado anterior. No caso das LGBM, o objetivo é combinar vários modelos de baixa performance num modelo de performance elevada. Para tal, cada novo classificador é treinado para reduzir os erros cometidos pelo anterior.
A escolha deste classificador visa integrar uma tecnologia que é recente, inovadora e já com resultados que comprovam a sua boa performance. Foi implementado recorrendo à framework *LightGBM*[24] da *Microsoft*, integrada com a API do *scikit-learn.*

7.6 Acesso à API

A plataforma web foi implementada com recurso à *framework Flask*[25], assente sob a linguagem Python. Esta *framework* apresenta uma forma conveniente de definir a API e de lidar com os pedidos e construção das respostas HTTP. Para que as várias funcionalidades do *Back Office* sejam acessíveis foram criados *endpoints.* Um *endpoint* é um ponto no canal de comunicação em que é possível aceder à API de forma a executar funções por ela permitidas.
No caso da configuração automática o *endpoint* pode ser acedido através do endereço "auto_conf/" com o método POST. O pedido enviado deve conter o nome a dar à configuração, o nome da base de dados a configurar, os seus dados de acesso e o nome do autor da configuração.
Já no caso da configuração manual cada *endpoint* permite aceder a uma componente da configuração escolhida. Caso o pedido seja feito com o método GET é retornado o estado dessa componente, caso seja feito com o método POST é criada essa componente com as propriedades definidas no pedido e caso seja feita com o método DELETE a componente é eliminada. A tabela seguinte resume as várias possibilidades de pedidos.

[24] Fonte: LightGBM
[25] Fonte: Flask

Tabela 19. *Endpoints relativos à configuração.*

ENDEREÇO	MÉTODOS	RESTRIÇÕES	DESCRIÇÃO
/auto_conf	POST	Utilizador deve ter sessão iniciada.	Permite criar uma nova configuração automática dada uma base de dados.
/users/<conf_id>	GET POST DELETE	Utilizador deve ter sessão iniciada. Os nomes de utilizador em cada configuração devem ser únicos.	Permite visualizar/adicionar/eliminar utilizadores dada uma configuração, nome, nome de utilizador e palavra-passe.
/tables/<conf_id>	GET POST DELETE	Utilizador deve ter sessão iniciada. As tabelas em cada configuração devem ser únicas.	Permite visualizar/adicionar/eliminar tabelas dada uma configuração, nome na BD, nome de interface, posição e tipo.
/attributes/<conf_id>	GET POST DELETE	Utilizador deve ter sessão iniciada. Os atributos em cada configuração devem ser únicos.	Permite visualizar/adicionar/eliminar atributos dada uma configuração, nome na BD, nome de interface, posição, tipo, tabela a que pertence e agregador predefinido.
/synonyms/<conf_id>	GET POST DELETE	Utilizador deve ter sessão iniciada.	Permite visualizar/adicionar/eliminar sinónimos dada uma configuração, atributo e sinónimo.
/auto_completes/<conf_id>	GET POST DELETE	Utilizador deve ter sessão iniciada.	Permite visualizar/adicionar/eliminar uma sugestão de pesquisa dada uma configuração e o texto da sugestão a apresentar.
/system_config/<conf_id>	GET POST	Utilizador deve ter sessão iniciada.	Permite visualizar/modificar várias propriedades de uma configuração como a quantidade de dados a manter em memória, definir a data atual, mudar a linguagem, mudar as cores da interface e ativar/desativar o suporte para mapas.
/annotations/<conf_id>	GET POST DELETE	Utilizador deve ter sessão iniciada. Cada atributo tem no máximo uma anotação.	Permite visualizar/adicionar/eliminar anotações dada uma configuração, um atributo e o tipo de anotação.
/top_settings/<conf_id>	GET POST DELETE	Utilizador deve ter sessão iniciada.	Permite visualizar/adicionar/eliminar customização para pesquisas com a palavra "top" dada uma configuração, um atributo a medir, tipo de agregação e forma de ordenação.
/attribute_links/<conf_id>	GET POST DELETE	Utilizador deve ter sessão iniciada. Os atributos escolhidos devem estar previamente declarados.	Permite visualizar/adicionar/eliminar uma união entre várias tabelas (*LinkTable)* dada uma configuração e os dois atributos pelo qual as tabelas devem ser unidas.
/date_dimensions/<conf_id>	GET POST DELETE	Utilizador deve ter sessão iniciada.	Permite visualizar/adicionar/eliminar um calendário fiscal a uma configuração.

/feedback/<conf_id>	GET POST DELETE	Utilizador deve ter sessão iniciada.	Permite visualizar/adicionar/eliminar feedback a uma configuração, com um autor, pontuação de 0 a 5 e descrição associados.
/save_to_ansa	POST	Utilizador deve ter sessão iniciada.	Permite traduzir e escrever na base de dados do ANSA a configuração a pôr em prática.

Capítulo 8
Testes

Neste capítulo são descritos e apresentados os testes ao protótipo final. Numa primeira fase, são testados os classificadores candidatos a integrar a configuração automática e, de seguida, são testados os resultados dessa configuração automática com os classificadores escolhidos.

8.1 Classificação

Metodologia

Para escolher o classificador a ser integrado na configuração automática, foram realizados vários testes comparativos entre eles. Inicialmente, foram testados os classificadores baseado em regras, que não requerem um tratamento dos dados tão exigente como os modelos de aprendizagem computacional. Já para estes, foi usado o pré-processamento dos dados referido na secção 7.5.

Como referido anteriormente na descrição dos dados, estamos perante um *dataset* desbalanceado. Como tal, foi utilizada validação cruzada estratificada para o treino dos modelos. Esta técnica utiliza o mesmo procedimento da validação cruzada tradicional, que divide o conjunto total de dados em *k* subconjuntos mutuamente exclusivos e do mesmo tamanho. De seguida, um desses subconjuntos é utilizado para teste e os restantes para treino. Este processo repete-se *k* vezes alternando de forma circular o subconjunto de teste. Na validação cruzada estratificada, para além do procedimento referido, existe ainda a garantia de que cada um destes subconjuntos contém aproximadamente a mesma percentagem de amostras de cada classe do conjunto completo. O valor de *k* usado foi 5.

Para os classificadores que o suportam foi usada a atribuição de pesos às diferentes classes de forma a garantir que o desbalanceamento do *dataset* não invalida os resultados.

Foi também testada a discretização dos dados, reduzindo os dados contínuos em 90%.

Relativamente às técnicas de redução de características, o método *SelectKBest* permite escolher as *K* melhores características a manter, parâmetro que foi testado com os valores 50, 100 e 200.

O método PCA permite personalizar a percentagem da variância a manter dos dados originais. Este parâmetro foi testado com os valores 90% e 95%. Por último, o método LDA não tem parâmetros personalizáveis e foi também testado o caso em que não é utilizado qualquer método de redução de características.

Relativamente aos modelos de aprendizagem computacional, cada um foi testado com os seus hiperparâmetros mais comuns. No caso do KNN, foi variado o valor de "*K",* isto é, o número de vizinhos a considerar na classificação, a variável "peso" que define a importância que é dada a cada vizinho e ainda a métrica pela qual as distâncias são avaliadas.

Para o modelo SVM foram variados os parâmetros "*C"* que determina a influência das previsões erradas, "*gamma"* que influencia a curvatura no hiperplano de decisão e ainda

a função de *kernel*. Este classificador suporta o balanceamento de classes através da atribuição de pesos, mecanismo este que foi testado.
O facto dos modelos RF e LGBM serem ambos baseados em árvores de decisão faz com que partilhem alguns hiperparâmetros entre si. Estes hiperparâmetros são o número de estimadores, a profundidade da árvore e o número de folhas. No caso do LGBM também foram testados os valores de taxa de aprendizagem e variado o tipo de *boosting*. Tal como no SVM, foi testada a atribuição de pesos às classes em ambos os modelos.
A Tabela 20 apresenta os intervalos de valores testados.

Tabela 20. Tabela com todos os valores testados.

FASE	MÉTODO	PARÂMETROS
Discretização	*KBinsDiscretizer*	Redução de valores distintos: [*None*, 90%]
Redução de características	Nenhuma	----------
	LDA	----------
	PCA	Variância preservada: [90%, 95%]
	SelectKBest	Nº de características: [50, 100, 200]
Classificação	*KNN*	Vizinhos: [1, 5, 11, 51, 101] Pesos: [uniforme, por distância] Métricas: [Euclidiana, Manhattan]
	SVM	C: [0.1, 1, 10] *Gamma:* [‘*scale*’, ‘*auto*’] *Kernel:* [‘rfb’] Pesos por classe: [balanceado]
	RF	Nº de estimadores: [100, 200] Profundidade máxima: [*None*, 100, 200] Nº de amostras mínimas para nós internos: [2, 30, 50] Nº de amostras mínimas para folhas: [1, 10, 50] Pesos por classe: [balanceado, balanceado sub-amostras]
	LGBM	Tipo de *boosting*: [‘gbdt’, ‘dart’, ‘goss’] Taxa de aprendizagem: [0.1, 1] Profundidade máxima: [*None*, 10, 50] Nº de folhas: [31, 100, 200] Nº de estimadores: [50, 100, 200] Pesos por classe: [balanceado]

Para automatizar a pesquisa dos parâmetros a testar recorreu-se à classe *GridSearch*[26] que testa cada classificador de forma exaustiva com as opções de pré-processamento e hiperparâmetros descritas anteriormente.

[26] Fonte: sklearn.model_selection.GridSearchCV — scikit-learn documentation

Métricas

Antes dos vários resultados poderem ser analisados, foi ainda necessário decidir a métrica de performance pelo qual vão ser comparados. As métricas mais comuns são a exatidão (*accuracy*), precisão (precision), abrangência (*recall*) e *f1-score.*

A exatidão é a métrica mais facilmente interpretável. Ela diz-nos a percentagem de exemplos corretamente classificados, independentemente da classe a que pertencem. A sua fórmula é a seguinte:

$$Exatidão = \frac{Verdadeiros\ Positivos + Verdadeiros\ Negativos}{Verdadeiros\ Positivos + Falsos\ Positivos + Verdadeiros\ Negativos + Falsos\ Negativos}$$

Figura 32 - Fórmula exatidão

A precisão é a métrica que nos permite medir a quantidade de casos positivos de todos os casos positivos previstos. Logo, a precisão é útil quando o custo de falsos positivos é elevado. A sua fórmula é a seguinte:

$$Precisão = \frac{Verdadeiros\ Positivos}{Verdadeiros\ Positivos + Falsos\ Positivos}$$

Figura 33 - Fórmula precisão

A abrangência é a métrica que nos permite medir a quantidade de casos previstos como positivos de todos os casos positivos existentes. Logo, a abrangência é útil quando o custo de falsos negativos é elevado. A sua fórmula é a seguinte:

$$Abrangência = \frac{Verdadeiros\ Positivos}{Verdadeiros\ Positivos + Falsos\ Negativos}$$

Figura 34 - Fórmula abrangência

Por último, temos os *f1-score* que é a média harmónica entre a precisão e a abrangência definidas anteriormente. Isto leva a que, caso a precisão ou a abrangência sejam baixas, o *f1-score* também o seja. Assim, esta métrica dá-nos o balanço entre falsos positivos e falsos negativos. A sua fórmula é a seguinte:

$$F1 - score = 2 * \frac{(Precisão * Abrangência)}{(Precisão + Abrangência)}$$

Figura 35 - Fórmula *f1-score*

Em ambos as abordagens aos problemas de classificação estamos perante conjuntos de dados desbalanceados, o que nos leva a considerar maioritariamente o *f1-score* como métrica de performance já que este tem em conta a forma como os dados estão distribuídos.

Resultados abordagem de classificação única

Para cada classificador testado o gráfico seguinte apresenta a melhor performance obtida na abordagem de classificação de dados única (tipo e anotações).

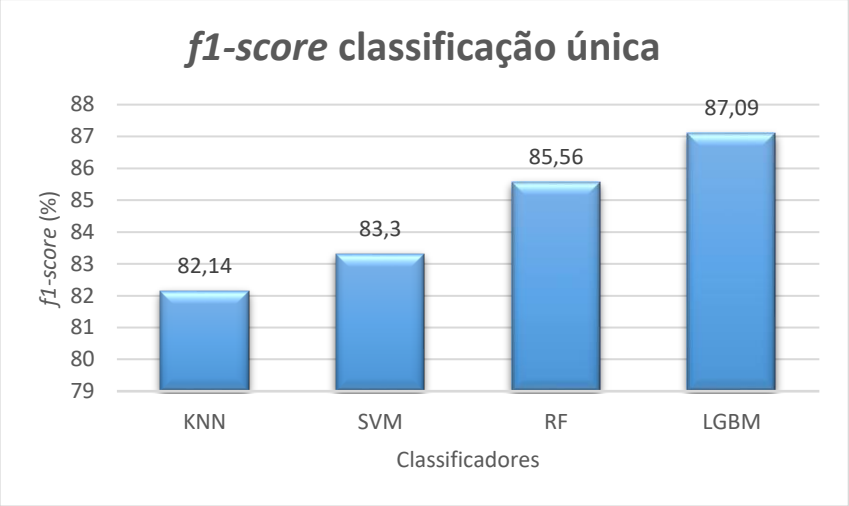

Figura 36 - Histograma com os melhores resultados obtidos na abordagem de classificação única.

Para além da performance foram também medidos os tempos de treino dos vários modelos, comparados na Figura 35.

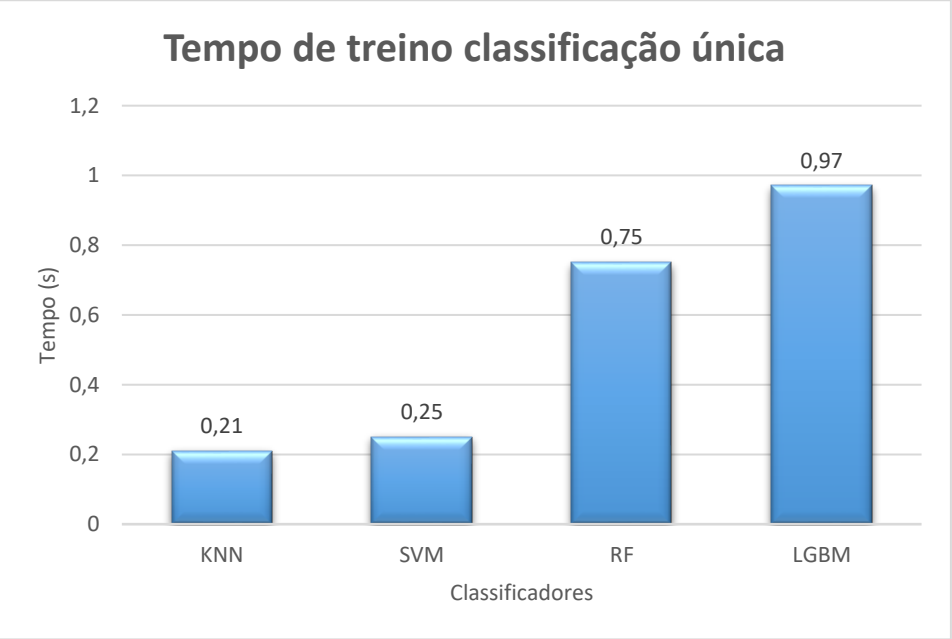

Figura 37 - Tempos de treino relativos à abordagem de classificação única

Analisando o gráfico de performance vemos que existe uma melhoria incremental desde o classificador mais simples KNN até ao mais complexo LGBM. Como seria de esperar, os métodos ensemble são aqueles que necessitam de mais tempo para serem treinados. Ainda assim, os tempos de treino mantêm-se todos abaixo de um segundo, fazendo com que este fator não tenha grande influência na escolha final. Assim, nesta

abordagem o classificador a escolher para integrar o protótipo final é o LGBM. Os melhores parâmetros encontrados para este classificador estão apresentados na Tabela 21:

Tabela 21. Tabela com os parâmetros de melhor performance para as RF na abordagem de classificação única.

FASE	MÉTODO	PARAMETROS
Discretização	*KBinsDiscretizer*	Redução de valores distintos: *None*
Redução de características	*SelectKBest*	Nº de características: 100
Classificação	*LGBM*	Tipo de *boosting*: 'dart'
		Taxa de aprendizagem: 0.1
		Profundidade máxima: 10
		Nº de folhas: 100
		Nº de estimadores: 200

Resultados abordagem de classificação em série

Relativamente à segunda abordagem testada, a abordagem em série, foram obtidos os seguintes *f1-scores*. A azul encontram-se os resultados da classificação quanto ao tipo de dados e a verde os resultados da classificação quanto a anotações.

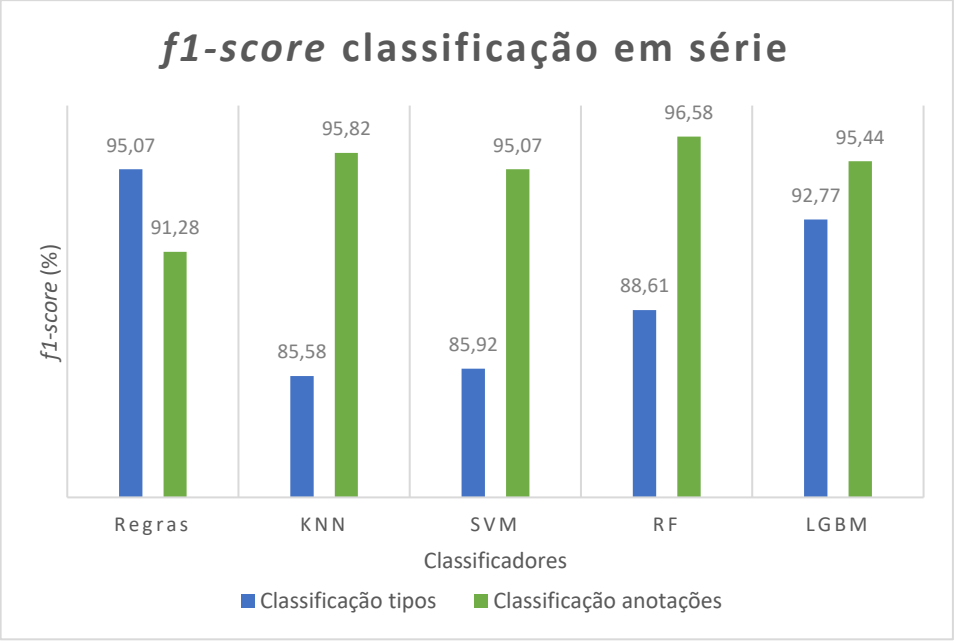

Figura 38 - Histograma com os melhores resultados obtidos na abordagem de classificação em série.

Mais uma vez, foram medidos os tempos de treino dos vários modelos, comparados na Figura 39.

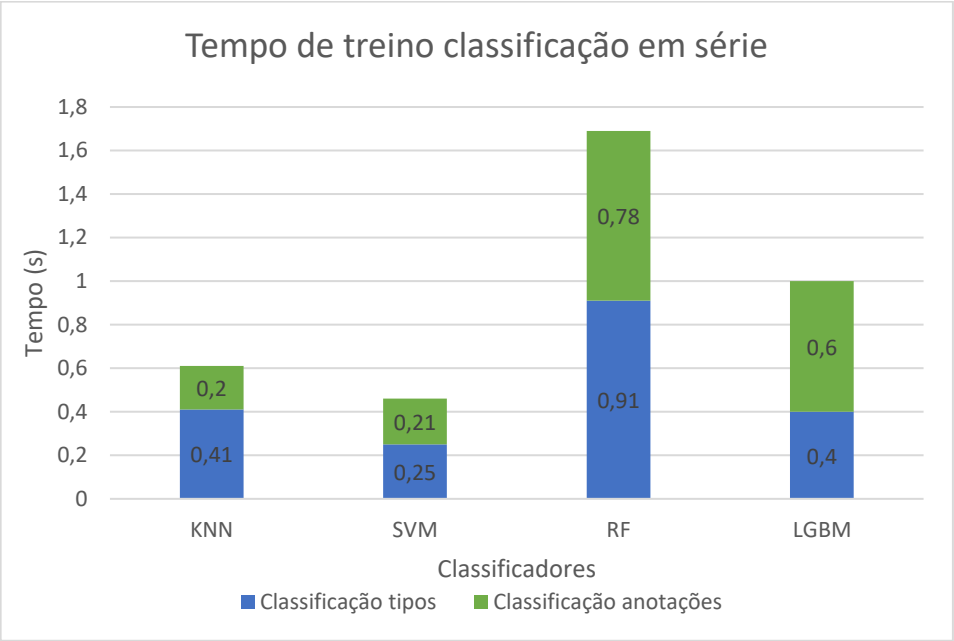

Figura 39 - Tempos de treino relativos à abordagem de classificação em série

Tal como na primeira abordagem, os tempos de treino para cada classificação mantêm-se abaixo de um segundo. Assim, este não deve ser o fator mais descriminante na escolha do classificador.

Relativamente às performances, há uma vantagem clara das regras na classificação quanto ao tipo de dados, o que, aliado ao facto de não necessitarem de treino, as tornam na opção mais viável. Já na classificação quanto às anotações, apesar do *f1-score* dos modelos de aprendizagem computacional ser muito semelhante, o melhor classificador são as RF. Concluindo, nesta abordagem a melhor alternativa é recorrer numa primeira classificação às regras e, posteriormente, às RF para classificação de anotações. Os melhores parâmetros encontrados para o classificador RF estão esquematizados na Tabela 22:

Tabela 22. Tabela com os parâmetros de melhor performance para as RF na abordagem de classificação em série.

FASE	MÉTODO	PARAMETROS
Discretização	*KBinsDiscretizer*	Redução de valores distintos: *None*
Redução de características	*SelectKBest*	Nº de características: 50
Classificação	*RF*	Nº de estimadores: 100
		Profundidade máxima: *None*
		Nº de amostras mínimas para nós internos: 2
		Nº de amostras mínimas para folhas: 10
		Pesos por classe: balanceado

Resumo

Analisando os dados de uma forma global podemos concluir que, em ambas as abordagens, os classificadores que recorrem a métodos *ensemble* são os que necessitam de mais tempo de treino, como seria de esperar. Ainda assim, todos os classificadores podem ser treinados em menos de um segundo. Como o âmbito da classificação não é de natureza crítica e a diferença nos tempos de treino não é significativa tendo em conta a perceção de tempo do ser humano, este fator não se revela decisivo para a escolha da melhor abordagem.

Analisando a performance, podemos concluir que a classificação em série é superior à classificação única e que os modelos a implementar no protótipo final devem ser o classificador baseado em regras para classificar os dados quanto ao seu tipo e o classificador RF para, de seguida, classificar os dados quanto à presença de anotações.

Esperava-se que os classificadores fossem capazes de distinguir colunas numéricas que devem ser consideradas como nominais (como explicado na secção 7.5) mas tal não aconteceu e o simples raciocínio aplicado pelas regras mostrou melhor performance.

8.2 Configuração automática

Os testes à configuração automática têm como objetivo testar a comunicação e a tradução das configurações geradas pelo *Back Office* no ANSA e são realizados com os classificadores que obtiveram melhor performance. Para que um teste seja considerado bem-sucedido a configuração deve ser iniciada no ANSA sem erros, com todas as tabelas e colunas da base de dados original presentes e a capacidade de fazer pesquisas com vários atributos e tabelas (caso existam).

A imagem seguinte ajuda a ilustrar uma configuração bem-sucedida, neste caso para um conjunto de dados relativos às eleições legislativas portuguesas, onde à esquerda podemos ver todos os atributos configurados e as pesquisas já realizadas e ao centro os resultados dessas pesquisas. Obter uma interface como a descrita na Figura 40 significa que o ANSA iniciou sem erros e que a configuração gerada pelo *Back Office* é válida permitindo aceder a todos os atributos e agregando atributos de tabelas diferentes (nos casos em que existe mais do que uma tabela).

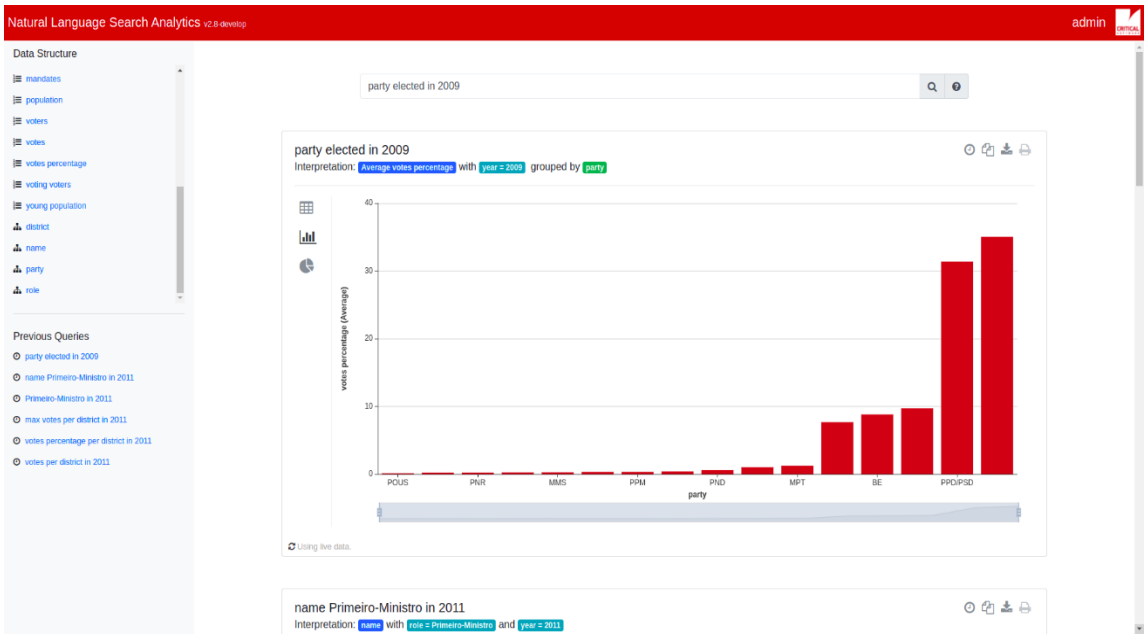

Figura 40 – Interface do ANSA com pesquisas realizadas sobre a configuração de eleições legislativas portuguesas.

Nas configurações em que existem anotações, pesquisas com estas anotações também foram testadas. A Figura 41 demonstra a anotação de país em funcionamento, configurada automaticamente pelo *Back Office*.

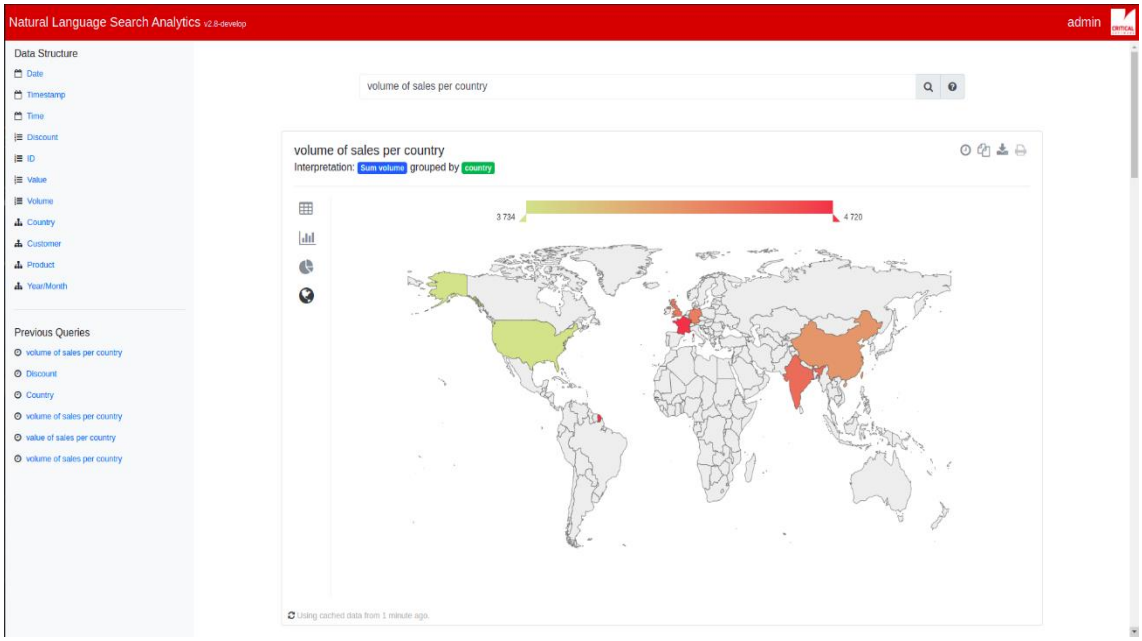

Figura 41 – Interface do ANSA testando a anotação de países numa configuração com dados de vendas.

Todos os conjuntos de dados disponibilizados pela CSW, juntamente com as configurações adicionadas durante o decorrer deste estágio foram testadas e os resultados são apresentados na tabela seguinte.

Tabela 23. Resultados dos testes às configurações

BASE DE DADOS	DESCRIÇÃO	RESULTADO
HOMEBANKING	Dados simulados de várias contas bancárias.	✔
JIRA	Dados retirados da plataforma de gestão de tarefas Jira, com apenas uma tabela	✔
PULSAR	Dados retirados da plataforma de gestão de recursos humanos Pulsar, com apenas uma tabela	✔
SALES	Dados simulados de vendas de uma empresa.	✔
SALES-NEW	Novos dados simulados de vendas de uma empresa.	✔
OVERSEE	Dados de ocorrências resolvidas pela marinha irlandesa. Múltiplas tabelas.	✔
COVID-19	Dados relativos à pandemia Covid-19.	✔
PT-ELECTIONS	Dados relativos às eleições legislativas portuguesas. Múltiplas tabelas.	✔
PURCHASES	Dados simulados de uma plataforma de vendas online.	✔

Capítulo 9
Conclusão

Durante o primeiro semestre foi adquirido o conhecimento técnico necessário mais tarde à implementação, foram conhecidas as ferramentas ao dispor e os produtos semelhantes existentes no mercado e definidos, juntamente com o cliente que neste caso é a CSW, os requisitos para o protótipo final. Depois de definidos os requisitos, recorreu-se a protótipos, diagramas de componentes, de entidades e de interação para apresentar uma proposta mais concreta daquilo que viria a ser o produto, e fizeram-se ajustes com o cliente de forma a garantir um alinhamento com as suas expetativas. Durante o segundo semestre o trabalho foi focado na implementação dos objetivos estipulados criando a plataforma web *Back Office* capaz de gerir as várias configurações criadas de forma automática ou manual.

Fazendo um balanço subjetivo da experiência, a maior dificuldade foi a interpretação da plataforma ANSA, já que este é um projeto realizado por várias pessoas, com uma dimensão e complexidade superiores aquilo que estava habituado do meio académico. Aponto como ponto negativo o facto de o trabalho ter de ser realizado de forma remota devido às medidas implementadas face à pandemia Covid-19, já que experienciar um ambiente empresarial na sua essência total era importante para mim. Por outro lado, como pontos positivos saliento o acompanhamento, comunicação e feedback providenciados por todos os orientadores, bem como o conhecimento adquirido naquilo que considero uma temática interessante e útil para o meu futuro.

Sendo o PLN uma área ainda não abordada aprofundadamente durante o mestrado, o trabalho realizado permitiu-me alargar os meus conhecimentos e descobrir novas tarefas e soluções possíveis de alcançar neste ramo da inteligência artificial. As várias componentes do protótipo trouxeram-me conhecimento relativo ao planeamento e implementação de sistemas, à comunicação entre plataformas web, à classificação de dados e às ferramentas como Docker, Jira e GitHub, indispensáveis a qualquer equipa nos dias de hoje.

Como trabalho futuro aponto a criação de uma interface que integre as funcionalidades implementadas e a adição de mais amostras ao conjunto de dados utilizado para treino dos classificadores. Apesar de terem sido utilizados todos os dados disponíveis atualmente, espera-se que com a adição de mais configurações seja possível elevar a performance dos modelos de aprendizagem computacional e que estes aprendam as nuances de como distinguir melhor as classes “latitude” e “longitude” e interpretar dados numéricos com poucos valores distintos como fazendo parte da classe “nominal”.

Referências

Accuracy vs. F1-Score. A comparison between Accuracy and... | by Purva Huilgol | Analytics Vidhya | Medium, from https://medium.com/analytics-vidhya/accuracy-vs-f1-score-6258237beca2

Atallah, D. M., Badawy, M., & El-Sayed, A. (2019). Intelligent feature selection with modified K-nearest neighbor for kidney transplantation prediction. SN Applied Sciences, 1(10). https://doi.org/10.1007/S42452-019-1329-Z

Ng, Andrew. Aprendizagem Automática | Coursera. Standford University., from https://pt.coursera.org/learn/machine-learning?

Bensouda Mourri, Y., Kaiser, Ł., & Shyu, E. (n.d.). Processamento da Linguagem Natural | Coursera. DeepLearning:AI., from https://pt.coursera.org/specializations/natural-language-processing

Cross-validation: evaluating estimator performance — scikit-learn 0.24.2 documentation, from https://scikit-learn.org/stable/modules/cross_validation.html#stratified-k-fold

Ensemble methods — scikit-learn 0.24.2 documentation, from https://scikit-learn.org/stable/modules/ensemble.html#forest

Feature selection — scikit-learn 0.24.2 documentation, from https://scikit-learn.org/stable/modules/feature_selection.html

Goldberg, Y. (2017). Neural Network Methods for Natural Language Processing. Synthesis Lectures on Human Language Technologies, 10(1), 1–311.

Gradient Boosting Classification explained through Python | by Vagif Aliyev | Towards Data Science, from https://towardsdatascience.com/gradient-boosting-classification-explained-through-python-60cc980eeb3d

Grishman, R., & Sundheim, B. (1996). Message Understanding Conference-6. 466.

Han, J., Kamber, M., & Pei, J. (2019). Data Mining, Concepts and Techniques. In Journal of Chemical Information and Modeling (Vol. 53, Issue 9).

KNN —K-Nearest Neighbor, o que é? | by Arthur Lamblet Vaz | Data Hackers | Medium, from https://medium.com/data-hackers/knn-k-nearest-neighbor-o-que-é-aeebe0f833eb

Lafferty, J., Mccallum, A., Pereira, F. C. N., & Pereira, F. (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data.

Linear and Quadratic Discriminant Analysis — scikit-learn 0.24.2 documentation, from https://scikit-learn.org/stable/modules/lda_qda.html#lda-qda

Liu, Z., Lin, Y., & Sun, M. (2020). Representation Learning for Natural Language Processing. In Representation Learning for Natural Language Processing.

Marshall, C. (2019). What is named entity recognition (NER) and how can I use it? | by Christopher Marshall | super.AI | Medium. Medium.

Mikheev, A., Moens, M., & Grover, C. (1999). Named Entity recognition without gazetteers. Proceedings of the Ninth Conference on European Chapter of the Association for Computational Linguistics .

Mills, T. (2017). What Is Natural Language Processing And What Is It Used For? Forbes.
Named Entity Recognition with NLTK and SpaCy | by Susan Li | Towards Data Science. (n.d.), from https://towardsdatascience.com/named-entity-recognition-with-nltk-and-spacy-8c4a7d88e7da
O que é o PCA - Principal Component Analysis | BI4ALL, from https://www.bi4all.pt/noticias/blog/o-que-e-o-pca/
Pires, A. R. O. (2017). Named entity extraction from Portuguese web text.
Ponomareva, N., Rosso, P., Pla, F., & Molina, A. (2007). Conditional Random Fields vs. Hidden Markov Models in a biomedical named entity recognition task. International Conference Recent Advances in Natural Language Processing, RANLP, 2007-Janua(May 2014), 479–483.
Poroshin, V. A. (2004). Semantic analysis of Natural Language. 14.
Python microservices with Flask · Viktor Adam's blog. (n.d.), from https://blog.viktoradam.net/2017/12/16/python-microservices-with-flask/
Random Forest Algorithms: A Complete Guide | Built In, from https://builtin.com/data-science/random-forest-algorithm
Rau, L. (1991). Extracting company names from text.
Setlur, V., Tory, M., & Djalali, A. (2018). The case for search-driven analytics: How a new breed of search engine is solving bi's adoption problem. ThoughtSpot.
Sharma, D. M. (2010). On the Role of NLP in Linguistics. Proceedings of the 2010 Workshop on NLP and Linguistics: Finding the Common Ground, July, 18–21.
StandardScaler — scikit-learn 0.24.2 documentation, from https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html
Support Vector Machine — Introduction to Machine Learning Algorithms | by Rohith Gandhi | Towards Data Science, from https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47
Tan, H., & Lambrix, P. (2009). Selecting an Ontology for Biomedical Text Mining.
Using KBinsDiscretizer to discretize continuous features — scikit-learn 0.24.2 documentation, from https://scikit-learn.org/stable/auto_examples/preprocessing/plot_discretization.html
Wen, Y., Fan, C., Chen, G., Chen, X., & Chen, M. (2020). A Survey on Named Entity Recognition. Lecture Notes in Electrical Engineering, 571 LNEE(1991), 1803–1810.