IPBeja

INSTITUTO POLITÉCNICO DE BEJA

Sistema de apoio à promoção do turismo rural
Fase de Investigação

Gonçalo Amaro – 17440,
Pedro Tomás – 18962,
Vítor Abreu – 18966

11 de Novembro, 2021

Conteúdo

1 Caracterização do objetivo

1.1 Objetivo do trabalho

O objetivo deste trabalho é a análise de dados de plataformas online relacionados com turismo, para o desenvolvimento do turismo rural alentejano. Isto é, obter todos os dados de *posts* e comentários relacionados com *providers* de acesso, entretenimento, refeições e estadia diretamente ligados ao património cultural do alentejo. Estes a serem analisados e classificados, criando assim um modelo de possíveis sentimentos e procuras que o comércio local tem o interesse em fornecer aos visitantes.

O nosso foco principal é nos *posts* e comentários em português principalmente, com a possibilidade de analisar também o conteúdo em inglês e fazer *cross-referencing*.

1.2 Objetivo desta investigação

Com esta investigação queremos obter o nosso curso de ação. Quais são os websites e os métodos de obtenção de dados, e os métodos de analisar os mesmos. Após exaustiva procura pelos websites preferenciais e discussão entre todos os elementos do grupo, para a utilização de um conjunto deles onde fosse possível se realizar uma boa análise e obtenção dos dados, com especial foco na linguagem portuguesa, reunimos alguns possíveis candidatos que serão abordados no ponto seguinte.

Foi decidido também que dentro das opções que serão referidas em seguida, teremos preferência na utilização de um em específico, uma vez que para alem da língua portuguesa temos intenções de dar alguma prioridade ás informações relacionadas com o turismo rural alentejano, sendo assim e devido a uma maior procura e número de resultados que o website oferecia, inicialmente iriamos utilizar o website TripAdvisor como primeira opção.

2 Sites pesquisados

Foram usados os sites:

- TripAdvisor
- Booking
- Zomato
- Google Maps

O nosso foco principal é o TripAdvisor visto que este oferece a maior variedade de conteúdo (hotéis, restaurantes e outros estabelecimentos), no entanto como uma plataforma é pouco, decidimos adicionar Booking e Zomato à lista para uma maior e mais ampla rede de hotéis e restaurantes.

Foi também considerado o Google Maps, mas este apresentou um novo set de problemas que vão ser descritos já de seguida.

3 (Im)possibilidade de uso de APIs

Em nenhum dos sites testados foi observada uma facilidade na obtenção de acesso às suas APIs, apenas alguns (3/4) ofereceram acesso à documentação da(s) mesma(s) facilmente. A maioria requer um contacto, que foi tentado e continua sem resposta até hoje (contactos iniciados por Amaro, entre dia 26 e 29 de Out, e hoje sendo dia 11 de Nov).

Dentro dos sites que oferecem documentação foi observado que todos subdividem os seus serviços de API em 3 ou 4 APIs para casos de uso específicos (reservas, dados, etc) em vez de uma com *endpoints* que oferecem solução para todos os casos.

A anomalia aqui é o Google Maps que é o único que facilita o acesso à API (mas paga, temos de ver os créditos disponíveis no *Cloud Platform*), e de elevada dificuldade em *scraping* pelo óbvio.

4 Alternativas

Sendo que é impossível o uso das APIs (que facilitariam o trabalho) temos de recorrer a outras técnicas para obter os dados.

4.1 *Web-crawling* VS *web-scraping*

Web crawling, também conhecido como Indexação, é usado para indexar as informações na página usando bots também conhecidos como *trackers*. O *tracker* é essencialmente o que os motores de busca fazem. É uma questão de visualizar uma página como um todo e indexá-la. Quando um Bot rastreia um site, ele passa por todas as páginas e todos os links, até a última linha do site, em busca de qualquer informação.

O *web scraping*, também conhecido como extração de dados da *web*, é semelhante ao *web crawling*, pois identifica e localiza os dados de destino das páginas da *web*. A principal diferença é que, com o *web scraping*, sabemos quem identificou o conjunto de dados exatamente, por exemplo, uma estrutura de elemento HTML para páginas da *web* que estão a ser corrigidas, da qual os dados precisam ser extraídos.

Com isto visto, *web scraping* é o nosso alvo, visto que minimiza lixo e é direcionado. No entanto devemos olhar para:

4.1.1 Vantagens de *web scraping*

1. Mais rápido: É possível manusear grandes quantidades de dados que poderiam levar dias ou semanas a serem processados através do trabalho manual, com o uso do scraping podemos reduzir substancialmente o esforço e aumentar a velocidade de decisão.

2. Confiável e consistente: Ao fazer o trabalho manual é muito fácil de haver erros, por exemplo, erros tipográficos, informações esquecidas ou inserção nas colunas erradas. O uso do *web scraping* garante consistência e a qualidade dos dados.

3. Ajuda a reduzir a carga de trabalho.

4. Menor custo: Uma vez implementado o *scraping*, o custo total da extração de dados é significativamente reduzido, especialmente quando comparado ao trabalho manual.

5. Manutenção básica: Fazer o *scraping* de dados geralmente não requer muita manutenção.

4.1.2 Desvantagens de *web scraping*

1. Baixa proteção: Se os dados na *web* são protegidos, o uso do scraping também pode se tornar um desafio e aumentar os custos.

2. Dados estruturados: Não vai ser possível fazer scraping a 1000 sites diferentes pois cada site tem uma estrutura completamente diferente. Será necessário haver alguma estrutura básica que seja diferente em determinadas situações.

4.2 Necessidade de *Web Scraping*

Com o acima dito, é necessário recorrer a soluções de *Web Scraping*.

A comunidade internauta reparou no mesmo, visto que em reação ao observado existem dezenas de projetos e tutoriais de *Web Scraping* das variadas plataformas de turismo e reservas. Infelizmente, as mesmas não ajudam no processo e cada uma têm um forma de atuação bastante diferente.

5 Bibliotecas de Python para *Web Scraping*

Para fazer *Web Scraping* vamos usar Python, pela sua facilidade de uso e multifaceta *"Development speed is more important than execution speed"*.[1] Com Python também temos as opções de criar cadernos Jupyter onde o próprio código e os resultados são "encadernados" com parágrafos de texto fazendo o próprio projeto o seu pequeno relatório de progresso e resultados; como também a criação de ambientes virtuais (*containers*), onde os pacotes usados ficam registados e instalados localmente, garantindo assim a portabilidade.[2]

Para tal linguagem existem 5 grandes bibliotecas para a resolução deste caso:

- Requests
- BeautifulSoup
- Scrapy
- lxml

Cada uma tem diferentes vantagens e desvantagens. Caso nenhuma destas resultar, usaremos Selenium, que é uma biblioteca mais completa e poderosa que as listadas, visto que é uma ferramenta de testes de automação.[3] Porém tem um nível de complexidade maior e requer um setup inicial maior e mais trabalhoso, requer *WebDrivers* para a execução das tarefas, pode ser complicada com Firefox, sendo preferencial usar *Chromium-based Browsers* como o Chrome e o novo Edge (necessário ainda o *ChromeDriver* e o *EdgeDriver*)..[4] Tentaremos evitar essa, a todo o custo, pela sua complexidade e extras desnecessários às nossas necessidades e custo temporal do setup inicial.[5]

Para cada website pode ser necessário usar bibliotecas diferentes por necessidade ou por obtenção de informações/blog posts/etc. . . que facilitem ou melhorem o output desejado.[6]

5.1 Tratamento de output

Os outputs do conteúdo *scraped* podem vir em .xml ou .csv (ou outras mas essencialmente essas duas). Tentaremos ao máximo usar .csv, e transformar qualquer .xml em .csv, visto que um maior número de ferramentas gráficas (Excel, PowerBI, etc. . .) e/ou bibliotecas de Python (Pandas, matplotlib, etc) para a análise de dados tratam melhor ficheiros separados por vírgulas.

Será feita também uma análise e extração de *keywords* nos textos das descrições e *reviews*. Para tal existem variados algoritmos que podemos usar, alguns "clássicos"outros até de *machine learning*.

Inicialmente todas as informações (dados e meta-dados) são dados como relevantes, após consideração e ponderação durante análises iniciais do decorrer do estudo poderemos descartar dados que não consideremos relevantes. No entanto nada nos impede de tentar prever ou imaginar quais esses serão e posteriormente avaliar o nosso julgamento para ver o que foi aprendido.

5.1.1 Informações necessárias (previsão)

Qualquer dado diretamente relacionado com o estabelecimento como horas de funcionamento, localização, preços, classificação e quantidade de *feedback*. Também a relação das *keywords* apanhadas do *feedback* em relação ao oferecido.

5.1.2 Informações a descartar (previsão)

Provavelmente informações pessoais de clientes e empregados não nos serão minimamente importantes.

6 Análise

6.1 Algoritmos de mineração de texto

Os algoritmos de análise de texto podem ser considerados ferramentas de mineração de texto, isto é, o processo de descoberta de conhecimento potencialmente útil e inicialmente desconhecido, ou seja, a extração de conhecimento útil utilizando bases textuais.[7]

O processo de mineração de texto é dividido em quatro etapas bem definidas:

- Seleção
- Pré-processamento
- Mineração
- Assimilação

Na seleção, os documentos relevantes devem ser escolhidos e mais tarde processados. No pré-processamento ocorrerá a conversão dos documentos em uma estrutura compatível com minerador, bem como ocorrerá um tratamento especial do texto. Na mineração, o minerador irá detetar os padrões com base no algoritmo escolhido. E por fim, na assimilação, os utilizadores irão utilizar o conhecimento gerado para apoiar as suas decisões.[8]

A etapa pré-processamento pode ser dividida em quatro tarefas:

- Remover *StopWords*
- Compilação
- Normalização de sinónimos

- Indexação

Na etapa Remover *stopwords* os termos com pouca ou nenhuma relevância para o documento serão removidos. São palavras auxiliares ou conetivas, ou seja, não são discriminantes para o conteúdo do documento.[9]

Na etapa seguinte, compilação, realiza-se uma normalização morfológica, ou seja, as palavras são reduzidas ao seu radical, serão combinadas em uma única representação. A radicalização pode ser efetuada com o auxílio de algoritmos de radicalização, sendo os mais utilizados o algoritmo de Porter (Porter Stemming Algorithm) e algoritmo de Orengo (Stemmer Portuguese ou RLSP).[8]

Após a compilação, na etapa de normalização de sinónimos, os termos que possuem significados similares serão agrupados em um único termo, por exemplo, as palavras ruído, tumulto e barulho serão substituídas ou representadas pelo termo barulho.

E, por fim, na etapa indexação atribui-se uma pontuação para cada termo, garantindo uma única instância do termo no documento. No processo de atribuição de pesos devem ser considerados dois pontos:

- Quanto mais vezes um termo aparece no documento, mais relevante ele é para o documento
- Quanto mais vezes um termo aparece na coleção de documentos, menos importante ele é para diferenciar os documentos

6.2 Algoritmos Clássicos

Os algoritmos clássicos são instruções passo a passo de modo a que, dada uma entrada específica, é possível rastrear e determinar exatamente a saída.[10]

- Algoritmos clássicos especificam as regras exactas para encontrar a resposta geral
- Um algoritmo clássico usa código e dados para prever a resposta correta para uma pergunta
- Algoritmo clássico produz uma saída com base nas etapas descritas no algoritmo
- Em algoritmos clássicos, normalmente é necessário um grande número de exemplos para determinar a que distância a equação está da equação desejada. É por isso que "*big data*" é uma grande negócio hoje em dia
- Um algoritmo clássico não dá uma solução depois de chegar a uma solução optima para um problema

6.3 Algoritmos de *machine learning*

Os algoritmos de *machine learning* são partes de código que ajudam as pessoas a explorar, analisar e localizar o significado em conjuntos de dados complexos.

Os algoritmos de *machine learning* utilizam parâmetros baseados em dados de preparação, um subconjunto de dados que representa o conjunto maior. À medida que os dados de preparação se expandem para representar o mundo de forma mais realista, o algoritmo calcula resultados mais precisos.[10]

Algoritmos diferentes analisam os dados de diversas formas. Geralmente, são agrupados consoante as técnicas de *machine learning* para as quais são utilizados:

- Aprendizagem supervisionada
- Aprendizagem não supervisionada
- Aprendizagem por reforço

6.3.1 Aprendizagem supervisionada

Na aprendizagem supervisionada, os algoritmos fazem previsões com base num conjunto de exemplos etiquetados fornecidos por si. Esta técnica é útil quando sabe como deverá ser o resultado. Por exemplo, fornece um conjunto de dados que inclui populações de cidades por ano nos últimos 100 anos e deseja saber qual será a população de uma cidade específica dentro de quatro anos. O resultado utiliza etiquetas que já existem no conjunto de dados: população, cidade e ano.

6.3.2 Aprendizagem não supervisionada

Na aprendizagem não supervisionada, os pontos de dados não são etiquetados. O algoritmo etiqueta-os ao organizar os dados ou ao descrever a sua estrutura. Esta técnica é útil quando não sabe como deverá ser o resultado. Por exemplo, fornece dados de cliente e deseja criar segmentos de clientes que gostam de produtos semelhantes. Os dados que está a fornecer não são etiquetados e as etiquetas no resultado são geradas com base nas semelhanças descobertas entre os pontos de dados.

6.3.3 Aprendizagem de reforço

A aprendizagem por reforço utiliza algoritmos que aprendem com resultados e decide a ação a realizar em seguida. Após cada ação, o algoritmo recebe comentários que o ajudam a determinar se a escolha feita foi correta, neutra ou incorreta. Por exemplo, se estivermos a criar um carro autónomo, queremos que este cumpra a lei e mantenha as pessoas seguras. À medida que o carro ganha experiência e um histórico de reforço, aprende a permanecer dentro da faixa, a não ultrapassar o limite de velocidade e a travar quando encontrar peões.

Existem muitos tipos diferentes de algoritmos de *machine learning*.Contudo, por norma, os casos de utilização destes algoritmos enquadram-se numa destas categorias.

- Algoritmos de classificação de duas classes (binários) dividem os dados em duas categorias. São úteis para perguntas com apenas duas respostas possíveis mutuamente exclusivas, incluindo perguntas de sim/não
- Algoritmos de classificação multiclasse (multinomial) dividem os dados em três ou mais categorias. São úteis para perguntas com três ou mais respostas possíveis mutuamente exclusivas
- Algoritmos de deteção de anomalias identificam os pontos de dados que estão fora dos parâmetros definidos para o que é considerado "normal"
- Algoritmos de regressão preveem o valor de um novo ponto de dados com base em dados históricos
- Algoritmos de séries temporais mostram as alterações a um determinado valor ao longo do tempo. Com a análise e a previsão de série temporal, os dados são recolhidos a intervalos regulares ao longo do tempo e utilizados para fazer previsões e identificar tendências, sazonalidade, periodicidade e irregularidade
- Algoritmos de *clustering* dividem os dados por vários grupos ao determinar o nível de semelhança entre os pontos de dados
- Algoritmos de classificação utilizam cálculos de previsão para atribuir dados a categorias predefinidas

6.4 Decisão sobre o tipo de algoritmo

Após a grande análise dos tipos e subtipos de algoritmos de mineração de texto (análise de texto), é óbvia a escolha em algoritmos de machine learning com aprendizagem não supervisionada para a execução da nossa análise; a questão está em qual serão usados visto que muitos deles para os nossos casos poderão ter de ser sujeitos a pré-processamento; o que removeria as nossas requeridas dimensões de análise textual. No entanto vai continuar a haver pré-processamento como algoritmos de redução de dimensionalidade, a diferença comparado com a frase anterior é quão a pré-modelação não afetará negativamente os resultados e possíveis associações.

6.4.1 Algoritmos considerados

Dos variados algoritmos vistos e disponíveis na internet ou em bibliotecas de Python (como SciKitLearn, Tensorflow, Keras), tomamos a decisão de considerar os seguintes algoritmos como candidatos a uso e/ou pertencentes aos grupos de algoritmos usados para comparação de resultados:

- LDA (Latent Dirichlet Allocation): um modelo de distribuição gaussiana, muito usado por empresas de software sobre *feedback* e bug reports para associação de resultados do QA,
- K-Means Clustering: muito usado para fazer clusters de keywords em redes sociais,

- KNN (K-Nearest Neighbor): usado para agrupar dados relacionais com os contactos, relatórios, correspondência e emails em empresas,
- SVM (Support Vector Machines): este é usado nos mesmos lugares que regressões lineares, porém mais rápido ou poderoso, é usado para agrupar pontos como texto com imagens ou tópicos de texto em sites de vendas de 2ª mão.

No entanto existe um algoritmo de machine learning semi-supervisto (aprendizagem não supervista mas ele faz a sua auto-supervisão; logo é questão de semântica). Este é o:

- LSTM (*Long-short term memory*): que é um tipo de RNN (rede neural recorrente) com ou sem um *layer* CNN (um *layer* de convulsão).

Poderá ser usado este método visto que não só "está na moda" como existe muito material de apoio por esse motivo, isto se houver tempo extra para experimentar e se essa experiência produzir melhores resultados comparativamente a um SVM por exemplo. . . [11]

Pode-se notar que aqui foram escolhidos algoritmos de *machine learning* já comuns a grupos que realizaram tarefas semelhantes e que são algoritmos simples e rápido não sendo mais um algoritmo de uma família de algoritmos (mais complexos ou não, mas que têm muita variedade), tais como redes neurais (à exceção do RNN LSTM), algoritmos genéticos ou algoritmos lineares (como regressões lineares).

7 Webgrafia

[1] K. Parker, "Data science skills: Web scraping using python," Jan 2020. [Online]. Available: https://towardsdatascience.com/data-science-skills-web-scraping-using-python-d1a85ef607ed

[2] S. Li, "Web scraping tripadvisor, text mining and sentiment analysis for hotel reviews," Dec 2018. [Online]. Available: https://towardsdatascience.com/scraping-tripadvisor-text-mining-and-sentiment-analysis-for-hotel-reviews-cc4e20aef333

[3] W. Koehrsen, "Web scraping, regular expressions, and data visualization: Doing it all in python," May 2018. [Online]. Available: https://towardsdatascience.com/web-scraping-regular-expressions-and-data-visualization-doing-it-all-in-python-37a1aade7924

[4] R. Taracha, "Introduction to web scraping using selenium," Apr 2018. [Online]. Available: https://medium.com/the-andela-way/introduction-to-web-scraping-using-selenium-7ec377a8cf72

[5] D. Gray, "Better web scraping in python with selenium, beautiful soup, and pandas," Apr 2018. [Online]. Available: https://medium.com/free-code-camp/better-web-scraping-in-python-with-selenium-beautiful-soup-and-pandas-d6390592e251

[6] B. Carremans, "Sentiment analysis with text mining," Jan 2019. [Online]. Available: https://towardsdatascience.com/sentiment-analysis-with-text-mining-13dd2b33de27

[7] J. Korstanje, "Text mining for dummies: Text classification with python," Mar 2020. [Online]. Available: https://towardsdatascience.com/text-mining-for-dummies-text-classification-with-python-98e47c3a9deb

[8] Unknown, "Mineração de texto: Análise comparativa de algoritmos - revista sql magazine 138," 2020. [Online]. Available: https://www.devmedia.com.br/mineracao-de-texto-analise-comparativa-de-algoritmos-revista-sql-magazine-138/34013

[9] S. Valcheva, "Text mining algorithms list: Text classification categorization clustering," May 2020. [Online]. Available: https://www.intellspot.com/text-mining-algorithms/

[10] Unknown, "Classic algorithm vs ml algorithm," Aug 2019. [Online]. Available: https://www.geekboots.com/story/classic-algorithm-vs-ml-algorithm

[11] D. Subramanian, "Text mining in python: Steps and examples," Jun 2020. [Online]. Available: https://pub.towardsai.net/text-mining-in-python-steps-and-examples-78b3f8fd913b