INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO ESTADO DO ESPÍRITO SANTO
SISTEMAS DE INFORMAÇÃO

LUAN EIRIZ MASIOLI

OTIMIZAÇÃO DE ROTAS DE COMPRAS EM SUPERMERCADOS: UM ALGORITMO BASEADO EM APRENDIZAGEM POR REFORÇO

Cachoeiro de Itapemirim
2023

LUAN EIRIZ MASIOLI

OTIMIZAÇÃO DE ROTAS DE COMPRAS EM SUPERMERCADOS: UM ALGORITMO BASEADO EM APRENDIZAGEM POR REFORÇO

Trabalho de Conclusão de Curso apresentado à Coordenadoria do Curso de Sistemas de Informação do Instituto Federal do Espírito Santo, Campus Cachoeiro de Itapemirim, como requisito parcial para a obtenção do título de Bacharel em Sistemas de Informação.

Orientador: Prof. Dr. Ricardo Maróquio Bernardo

Cachoeiro de Itapemirim

2023

(Biblioteca do Campus Cachoeiro de Itapemirim)

M397o Masioli , Luan Eiriz .

Otimização de rotas de compras em supermercados: um algoritmo baseado em aprendizagem por reforço / Luan Eiriz Masioli . - 2023.
43 f. : il. ; 30 cm..

Orientador: Ricardo Maróquio Bernardo

TCC (Graduação) Instituto Federal do Espírito Santo, Campus Cachoeiro de Itapemirim, Sistemas de Informação, 2023.

1. Inteligência artificial . 2. Aprendizado do computador. 3. Supermercados - Automação. I. Bernardo, Ricardo Maróquio. II.Título III. Instituto Federal do Espírito Santo.

CDD: 006.3

Bibliotecário/a: Renata Lorencini Rizzi CRB6-ES nº 085

MINISTÉRIO DA EDUCAÇÃO
INSTITUTO FEDERAL DO ESPÍRITO SANTO
CAI - COORDENADORIA DO CURSO DE BACHARELADO EM SISTEMAS DE INFORMACAO

FOLHA DE APROVAÇÃO-TCC Nº 17 / 2023 - CAI-CCSI (11.02.18.01.08.02.13)

Nº do Protocolo: 23151.004352/2023-19

Cachoeiro De Itapemirim-ES, 15 de dezembro de 2023.

LUAN EIRIZ MASIOLI

OTIMIZAÇÃO DE ROTAS DE COMPRAS EM SUPERMERCADOS: UM ALGORITMO BASEADO EM APRENDIZAGEM POR REFORÇO

Trabalho de Conclusão de Curso apresentado à Coordenadoria do Curso de Sistemas de Informação do Instituto Federal do Espírito Santo, Campus Cachoeiro de Itapemirim, como requisito parcial para a obtenção do título de Bacharel em Sistemas de Informação.

Orientador: Prof. Dr. Ricardo Maroquio Bernardo

Aprovado em 11 de dezembro de 2023

COMISSÃO EXAMINADORA

Prof. Dr. Ricardo Maroquio Bernardo
Instituto Federal Do Espírito Santo
Orientador

Prof. Me. Cristiano da Silveira Colombo
Instituto Federal Do Espírito Santo

Prof. Me. Daniel Jose Ventorim Nunes
Instituto Federal Do Espírito Santo

(Assinado digitalmente em 15/12/2023 12:33)
CRISTIANO DA SILVEIRA COLOMBO
PROFESSOR DO ENSINO BASICO TECNICO E TECNOLOGICO
CAI-CCSI (11.02.18.01.08.02.13)
Matrícula: 1673901

(Assinado digitalmente em 15/12/2023 13:16)
DANIEL JOSE VENTORIM NUNES
PROFESSOR DO ENSINO BASICO TECNICO E TECNOLOGICO
CAI-CCLI (11.02.18.01.08.02.06)
Matrícula: 1918045

(Assinado digitalmente em 15/12/2023 12:19)
RICARDO MAROQUIO BERNARDO
PROFESSOR DO ENSINO BASICO TECNICO E TECNOLOGICO
CAI-CCSI (11.02.18.01.08.02.13)
Matrícula: 2152606

DECLARAÇÃO DO AUTOR

Declaro, para fins de pesquisa acadêmica, didática e técnico-científica, que este Trabalho de Conclusão de Curso pode ser parcialmente utilizado, desde que se faça referência à fonte e ao autor.

Cachoeiro de Itapemirim, 11 de Dezembro de 2023.

LUAN EIRIZ MASIOLI

Dedico este trabalho a todos que, direta ou indiretamente, contribuíram para que o mesmo fosse realizado.

AGRADECIMENTOS

Gostaria de agradecer especialmente ao meu professor orientador Dr. Ricardo Maróquio Bernardo pela oportunidade e acompanhamento neste trabalho, ao meu professor das disciplinas de Anteprojeto, Projeto de Diplomação I e Projeto de Diplomação II, Eros Estevão de Moura, à minha banca examinadora e à todos os demais professores do IFES Campus Cachoeiro de Itapemirim que contribuíram para minha jornada acadêmica. Gostaria de agradecer também à minha família, pelo apoio e pela compreensão aos meus momentos de ausência e aos meus colegas de classe pelo companheirismo durante todo o curso.

“A vida é o que fazemos dela. As viagens são os viajantes. O que vemos não é o que vemos, senão o que somos.”

Fernando Pessoa

RESUMO

O tema refere-se à aplicação de aprendizado de máquina utilizando *Q-Learning* para otimizar a rota de compras dos consumidores em um supermercado. Envolve a utilização de algoritmos que analisam a disposição dos produtos nas prateleiras e as preferências dos clientes, para então sugerir a melhor rota de compras em termos de um trajeto mais curto. O objetivo é tornar o processo de compras mais eficiente e conveniente para os clientes, aumentando a satisfação e fidelização. A pesquisa nessa área envolve a aplicação de técnicas de aprendizado de máquina, aprendizagem por reforço e inteligência artificial. O modelo obtido conseguiu encontrar a melhor rota de compras para os produtos selecionados em todas as execuções do algoritmo, com diferentes combinações de produtos selecionados.

Palavras-chave: otimização de rotas, aprendizagem por reforço, *Q-Learning*.

ABSTRACT

The topic refers to the application of machine learning using Q-Learning to optimize the shopping route of consumers in a supermarket. It involves the use of algorithms that analyze the arrangement of products on the shelves and the preferences of customers, to then suggest the best shopping route in terms of a shorter path. The goal is to make the shopping process more efficient and convenient for customers, increasing satisfaction and loyalty. The research in this area involves the application of machine learning techniques, reinforcement learning and artificial intelligence. The model obtained was able to find the best shopping route for the selected products in all executions of the algorithm, with different combinations of selected products.

Keywords: route optimization, reinforcement learning, Q-Learning.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SÍMBOLOS

∞ Infinito

α Letra grega alpha

γ Letra grega gamma

SUMÁRIO

1 INTRODUÇÃO

Segundo Moraes e Silva (2017), os mercados e supermercados utilizam diversas estratégias para influenciar o comportamento de compra dos clientes, incluindo a criação de uma atmosfera agradável, com iluminação adequada, música ambiente e aromas agradáveis. Essas estratégias têm como objetivo aumentar o tempo que os clientes passam no estabelecimento, o que pode levar a um aumento nas vendas. O estudo também destaca que a atmosfera do varejo pode influenciar a satisfação e a lealdade dos clientes, o que pode ter um impacto significativo no desempenho do negócio.

No contexto dos supermercados, a utilização da aprendizagem por reforço para encontrar a melhor rota de compras pode trazer benefícios significativos para os consumidores. Ao considerar a localização dos produtos e as preferências dos clientes, o programa pode gerar rotas otimizadas que economizam tempo e dinheiro para os consumidores, enquanto aumentam a eficiência do processo de compras para os supermercados.

Compreender as técnicas de persuasão utilizadas pelos estabelecimentos comerciais é fundamental para os consumidores realizarem suas compras de forma consciente e eficiente. Além disso, a proposta do trabalho em questão traz uma abordagem inovadora, utilizando ferramentas matemáticas e de inteligência artificial para auxiliar os consumidores a economizarem tempo e dinheiro nas compras. Ao mapear a localização dos produtos e criar rotas otimizadas, o programa pode ajudar os clientes a evitarem distrações e focarem nos itens que realmente precisam, sem se deixarem levar por estratégias de marketing. Com isso, espera-se que o projeto traga benefícios para os consumidores, ao melhorar a experiência de compra e aumentar a eficiência no processo.

1.1 PROBLEMA

Blessa (2003), adaptado por Silva e Arroyo (2013) traz que uma exposição bem feita de mercadorias, para o consumidor, facilita a compra e lembra necessidades; e, para o varejista, permite a valorização da loja, a fidelização dos clientes e o aumento

da lucratividade. Como o autor mencionou acima, a disposição dos produtos nas prateleiras influenciam no comportamento dos clientes, inclusive lembram necessidades que os clientes não tinham em mente previamente.

Segundo Camargo, Toaldo e Sobrinho (2009), em um supermercado, um *layout* deve, essencialmente, facilitar a logística, aumentar o tempo de permanência do cliente na loja e as vendas. Na elaboração de um *layout* em supermercados, faz-se necessário instigar o consumidor a conhecer todo o local, a fim de estimular a compra impulsiva. Características da atmosfera interna da loja como luminosidade, aromas, temperatura, largura dos corredores, cores e som podem se tornar elementos chaves para prender a atenção do consumidor, sendo que é fundamental que a loja tenha espaço para estacionamento, facilidade de acesso e proporcione segurança aos pedestres.

1.2 OBJETIVOS

Este trabalho tem o objetivo de desenvolver um algoritmo capaz de treinar um agente e aplicá-lo em um programa no qual irá traçar a melhor rota de compras em um supermercado. Tendo como base os produtos escolhidos previamente pelo usuário.

1.2.1 Objetivos Específicos

a) Treinar um agente utilizando aprendizagem por reforço;

b) Otimizar a rota de compras em um supermercado;

c) Auxiliar, por meio da melhor rota, o usuário a comprar apenas o que ele estava predisposto a levar naquele dia.

1.3 MOTIVAÇÃO DO AUTOR

A motivação para o desenvolvimento deste trabalho surgiu após uma demanda pessoal. O autor estava com dificuldades de encontrar produtos que o supermercado vendia e, por isso, ficava dando voltas e mais voltas em gôndolas, inclusive nas que já havia percorrido anteriormente. Foi aí que o autor idealizou a criação de um algoritmo que

fosse capaz de retornar a melhor rota de compras. Evitando assim que percorresse o supermercado e procurasse os produtos de maneira aleatória e levando mais tempo.

2 REFERENCIAL TEÓRICO

No capítulo anterior foram abordados a introdução, os objetivos do trabalho proposto e também a motivação do autor que idealizou a solução proposta. Este tratará sobre o referencial teórico utilizado no trabalho.

2.1 LINGUAGEM PYTHON

Segundo Borges (2014), a linguagem Python foi criada em 1990 por Guido van Rossum no Instituto Nacional de Pesquisa para Matemática e Ciência da Computação da Holanda (CWI), e tinha originalmente foco em usuários como físicos e engenheiros. Hoje, a linguagem é bem aceita na indústria por empresas de alta tecnologia como: Google, Yahoo, Microsoft e Disney.

De acordo com Raschka, Patterson e Nolet (2020), com seu foco central na legibilidade, o Python é uma linguagem de programação interpretada de alto nível, amplamente reconhecida por ser fácil de aprender, mas ainda capaz de aproveitar o poder de linguagens de programação de nível de sistema quando necessário. Além dos benefícios da própria linguagem, a comunidade em torno das ferramentas e bibliotecas disponíveis torna o Python particularmente atraente para trabalho em ciência de dados, aprendizado de máquina e computação científica. De acordo com uma pesquisa recente [1], a linguagem Python manteve sua posição no topo como a linguagem mais amplamente utilizada em 2019 para análise, ciência de dados e aprendizado de máquina.

Historicamente, uma ampla variedade de diferentes linguagens de programação e ambientes foram utilizados para possibilitar a pesquisa em aprendizado de máquina e o desenvolvimento de aplicações. No entanto, à medida que a linguagem Python, de propósito geral, experimentou um crescimento tremendo de popularidade dentro da comunidade de computação científica na última década, a maioria das bibliotecas mais recentes de aprendizado de máquina e aprendizado profundo agora são baseadas em Python (RASCHKA; PATTERSON; NOLET, 2020).

[1] Fonte: Entrevista do site KDnuggets que interrogou mais de 1800 participantes.

2.2 INTELIGÊNCIA ARTIFICIAL

Inteligência artificial (IA), como um subcampo da ciência da computação, concentra-se no design de programas de computador e máquinas capazes de realizar tarefas para as quais os humanos têm natural habilidade, incluindo compreensão de linguagem natural, compreensão de fala e reconhecimento de imagens (RASCHKA; PATTERSON; NOLET, 2020).

2.3 APRENDIZADO DE MÁQUINA

Este trabalho está inserido no cenário de Aprendizado de Máquina (*Machine Learning*), como Monard e Baranauskas (2003) explana: Aprendizado de Máquina é uma área da Inteligência Artificial (IA) cujo objetivo é o desenvolvimento de técnicas computacionais sobre o aprendizado bem como a construção de sistemas capazes de adquirir conhecimento de forma automática. Um sistema de aprendizado é um programa de computador que toma decisões baseado em experiências acumuladas através da solução bem sucedida de problemas anteriores. Os diversos sistemas de aprendizado de máquina possuem características particulares e comuns que possibilitam sua classificação quanto à linguagem de descrição, modo, paradigma e forma de aprendizado utilizado.

O aprendizado de máquina é aprender com os dados e fazer previsões e/ou decisões. Normalmente, categorizamos o aprendizado de máquina como aprendizado supervisionado, não supervisionado e por reforço. No aprendizado supervisionado, existem dados rotulados; no aprendizado não supervisionado, não há dados rotulados; e na aprendizagem por reforço, há *feedbacks* avaliativos, mas não há sinais supervisionados. Classificação e regressão são dois tipos de problemas de aprendizado supervisionado, com saídas categóricas e numéricas respectivamente (LI, 2017).

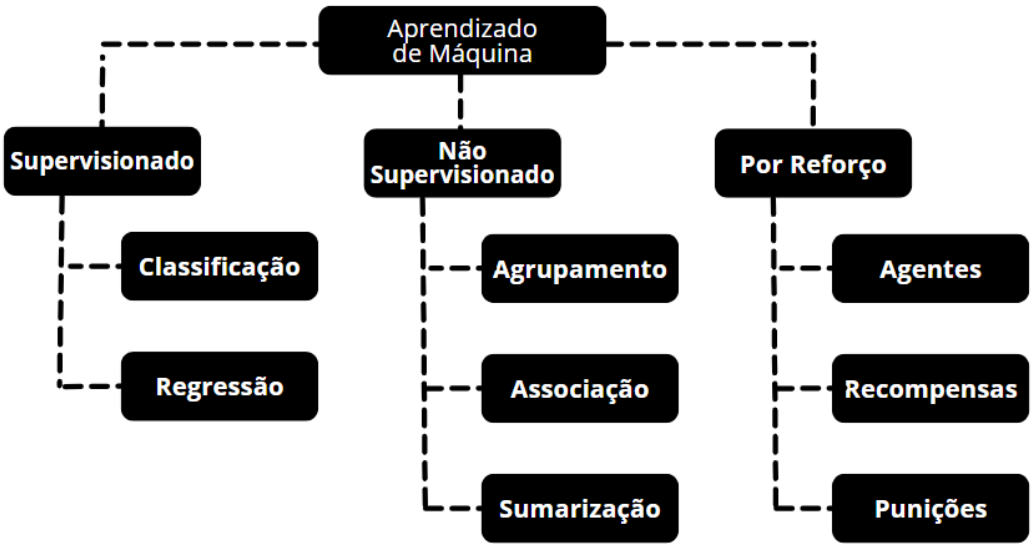

Figura 1 – Hierarquia do Aprendizado. Adaptado de Ludermir (2021)

2.3.1 Aprendizado Supervisionado

De acordo com Ludermir (2021) Aprendizado Supervisionado, para cada exemplo apresentado ao algoritmo de aprendizado é necessário apresentar a resposta desejada (ou seja, um rótulo informando a que classe o exemplo pertence, no caso de um problema de classificação de imagens, por exemplo, como distinguir imagens de gatos e de cachorros). Cada exemplo é descrito por um vetor de valores (atributos) e pelo rótulo da classe associada. O objetivo do algoritmo é construir um classificador que possa determinar corretamente a classe de novos exemplos ainda não rotulados. Para rótulos de classe discretos, esse problema é chamado de classificação e para valores contínuos como regressão. Também, segundo Ludermir (2021), este método de aprendizado é o mais utilizado.

A aprendizagem supervisionada é a aprendizagem a partir de um conjunto de treinamento de exemplos rotulados fornecidos por um supervisor externo qualificado. Cada exemplo é uma descrição de uma situação juntamente com uma especificação do rótulo da ação correta que o sistema deve tomar para essa situação, que muitas vezes consiste em identificar uma categoria à qual a situação pertence. O objetivo desse tipo de aprendizagem é que o sistema extrapole, ou generalize, suas respostas para que atue corretamente em situações não presentes no conjunto de treinamento. Este é um tipo importante de aprendizagem, mas por si só não é adequado para aprender a partir

da interação. Em problemas interativos é muitas vezes impraticável obter exemplos de comportamento desejado que sejam corretos e representativos de todas as situações em que o agente tem de agir. Em território desconhecido – onde se esperaria que a aprendizagem fosse mais benéfica – um agente deve ser capaz de aprender com a sua própria experiência (SUTTON; BARTO, 2018).

2.3.2 Aprendizado Não Supervisionado

No Aprendizado Não Supervisionado, os exemplos são fornecidos ao algoritmo sem rótulos. O algoritmo agrupa os exemplos pelas similaridades dos seus atributos. O algoritmo analisa os exemplos fornecidos e tenta determinar se alguns deles podem ser agrupados de alguma maneira, formando agrupamentos ou *clusters*. Após a determinação dos agrupamentos, em geral, é necessária uma análise para determinar o que cada agrupamento significa no contexto problema sendo analisado (LUDERMIR, 2021).

2.3.3 Aprendizagem por Reforço

Aprendizagem por reforço é um ramo estudado em estatística, psicologia, neurociência e ciência da computação. Atraiu o interesse de pesquisadores ligados a aprendizado de máquina e inteligência artificial, e é um método de programação de agentes através do oferecimento de recompensas e punições, sem a necessidade de especificar como uma tarefa deve ser realizada. É entendido como o problema encontrado por um agente que deve aprender como se comportar em um ambiente dinâmico através de interações do tipo “tentativa e erro” (JúNIOR, 2012).

A aprendizagem por reforço é diferente da aprendizagem supervisionada, o tipo de aprendizagem estudado na maioria das pesquisas atuais na área de aprendizado de máquina (SUTTON; BARTO, 2018). O aprendizado por reforço também é diferente do que os pesquisadores de aprendizado de máquina chamam de aprendizado não supervisionado, que normalmente consiste em encontrar estruturas ocultas em coleções de dados não rotulados. Os termos aprendizagem supervisionada e aprendizagem não supervisionada parecem classificar exaustivamente os paradigmas de aprendizagem de máquina, mas não o fazem. Embora alguém possa ficar tentado a pensar na aprendizagem por reforço como um tipo de aprendizagem não supervisionada porque não

se baseia em exemplos de comportamento correto, a aprendizagem por reforço tenta maximizar um sinal de recompensa em vez de tentar encontrar uma estrutura oculta. Descobrir a estrutura na experiência de um agente pode certamente ser útil na aprendizagem por reforço, mas por si só não resolve o problema da aprendizagem por reforço de maximizar um sinal de recompensa. Portanto, consideramos a aprendizagem por reforço um terceiro paradigma de aprendizagem de máquina, ao lado da aprendizagem supervisionada e da aprendizagem não supervisionada e talvez de outros paradigmas (SUTTON; BARTO, 2018).

2.3.3.1 Definição

Segundo Júnior (2012), em um ambiente de aprendizado por reforço, um agente está inserido em um ambiente T e interage com ele através de percepções e ações, conforme a Figura 2. A cada passo, o agente recebe como entrada E, uma indicação do estado (s) atual do ambiente. O agente escolhe, então, uma ação A a tomar, e gera sua saída. A ação altera então o estado do ambiente, e uma medida dessa mudança de estado é informada ao agente através de um valor de sinal de reforço, R. A função que mapeia os estados do ambiente nas ações que o agente deve tomar é definida como a política do agente. A política de comportamento do agente, P, deve escolher tomar ações que maximizem o valor final da soma dos reforços recebidos em um intervalo de tempo. Tal política de comportamento pode ser aprendida através de um processo de tentativa e erro, guiado por diferentes algoritmos.

Figura 2 – Modelo padrão de aprendizado por reforço (JúNIOR, 2012)

Na Figura 2 existe ainda a função de entrada e, que é a maneira como o agente “lê” o estado atual do ambiente.

A tarefa que o agente deve desempenhar é encontrar uma política π, que mapeia estados em ações, que maximiza a medida de reforço a longo prazo. Geralmente o sistema é não-determinístico; isso é, uma mesma ação tomada em um mesmo estado pode levar a diferentes estados, com diferentes valores de retorno percebidos. Ao contrário dos métodos conhecidos de aprendizado supervisionado, no aprendizado por reforço não existem pares “entrada/saída”, para serem utilizados no treinamento. Após tomar uma ação, o agente imediatamente recebe uma recompensa, mas não fica sabendo qual deveria ser a melhor ação para atingir o objetivo (maximizar o retorno a longo prazo). Ele precisa obter experiência dos possíveis estados, ações, transições e recompensas do sistema para atingir a otimalidade (JúNIOR, 2012).

2.3.3.2 Q-Learning

Segundo Watkins e Dayan (1992) *Q-Learning* é uma forma de aprendizagem por reforço. Também pode ser visto como um método de programação dinâmica assíncrona (DP). Ele fornece aos agentes a capacidade de aprender a agir de forma otimizada, experimentando o contexto e sequências de ações, sem exigir a construção de mapas dos domínios. Um agente tenta uma ação em um estado particular e avalia suas consequências em termos da recompensa ou penalidade imediata que recebe e sua estimativa do valor do estado para onde é levado. Ao tentar repetidamente todas as ações em todos os estados, ele aprende quais são as melhores no geral, avaliadas pela recompensa descontada a longo prazo. *Q-Learning* é uma forma primitiva de aprendizagem, mas, como tal, pode funcionar como base para dispositivos muito mais sofisticados. Existem também várias aplicações industriais.

Considere um agente computacional movendo-se em algum mundo discreto e finito, escolhendo uma dentre uma coleção finita de ações a cada passo de tempo. O mundo constitui um processo controlado, com o agente como controlador. Na etapa *n*, o agente está equipado para registrar o estado *x*, e pode escolher sua ação *a* de acordo. O agente recebe uma recompensa probabilística *Q*, dependendo apenas do estado e

da ação, de acordo com a lei:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)] \quad (1)$$

Onde, segundo Watkins e Dayan (1992):

a) Q(s,a) é o valor da ação *a* no estado *s*;

b) α é a taxa de aprendizado, que controla o quanto o agente atualiza seus valores a cada iteração;

c) *r* é a recompensa recebida pelo agente após executar a ação *a* no estado *s*;

d) γ é o fator de desconto, que controla o quanto o agente valoriza as recompensas futuras em relação às imediatas;

e) $\max_a Q(s_{t+1}, a) - Q(s_t, a_t)$ é o valor máximo da ação no próximo estado *s*.

2.4 ALGORITMO DE DIJKSTRA

O Algoritmo de Dijkstra é um dos algoritmos que calcula o caminho de custo mínimo entre vértices de um grafo. Escolhido o vértice A como raiz da busca (ponto de origem), este algoritmo calcula a distância mínima deste vértice para todos os demais vértices do grafo, ou seja, os demais pontos. Este algoritmo parte de uma estimativa inicial para a distância mínima, que é considerada infinita (∞), e vai sucessivamente ajustando esta distância (CARVALHO, 2008).

Este algoritmo poderia ser usado para resolver o problema deste trabalho, visto que ele é um algoritmo de pesquisa e solução em um gráfico para o caminho mais curto desde a origem ao destino (OLIVEIRA, 2015).

2.4.1 Definição

Em 1959, o holandês Edsger Dijkstra anunciou um algoritmo para resolver um problema conhecido como *single-source shortest path* (problema do caminho mínimo), cujo

objetivo era encontrar o menor caminho entre dois vértices de um grafo ponderado. Esse algoritmo é capaz de encontrar, por exemplo, a menor distância para uma viagem que atravessa várias cidades com diferentes caminhos para se chegar a um destino. Dado um vértice de início S, ele encontra o menor caminho a partir de S para todos os outros vértices no grafo, inclusive o destino T. Se as arestas são positivas, o algoritmo de Dijkstra tem bom desempenho. Uma implementação do algoritmo é executada num tempo O(m + n log n), onde n e m são número de vértices e arestas, respectivamente (MARTINS, 2015).

2.4.2 Funcionamento

Segundo Deng et al. (2012), para se obter o caminho mínimo de um vértice pertencente à um grafo utilizando o algoritmo de Dijkstra, deve-se executar os passos descritos a seguir. Assumindo que o vértice no início do caminho é o vértice de origem e que a distância de um vértice qualquer X será a distância do vértice de origem ao vértice X, o algoritmo atribuirá distâncias iniciais e tentará aprimorar a cada passo:

a) Atribuir um valor de distância a todos os vértices: zero para o vértice de origem e infinito para todos os outros vértices;

b) Marcar todos os vértices como não visitados. Estabelecer o vértice inicial (origem) como atual;

c) Para o vértice atual, considerar todos os vértices adjacentes e calcular uma tentativa de distância para cada um deles. Por exemplo, se o vértice atual A tem a distância de 6, e uma aresta o conecta a outro vértice B tem distância 2, a distância para B passando por A será 6 + 2 = 8. Se essa distância é menor que a distância registrada anteriormente, sobrescreva a distância;

d) Após calcular a distância para todos os vértices adjacentes ao vértice atual, marque-os como visitados. Um vértice que já foi visitado não será checado novamente; a distância registrada já é a mínima;

e) Se todos os vértices já foram visitados, parar. Caso contrário, estabelecer o vértice com menor distância para vértice inicial como vértice atual e retroceder ao passo 3.

A figura 3 ilustra a execução do algoritmo de Dijkstra. Os pesos das arestas estão indicados próximos às arestas e o valor da distância no momento da iteração está dentro do vértice. O número 99 representa infinito. Os vértices marcados como visitados estão grifados com cor verde (MARTINS, 2015).

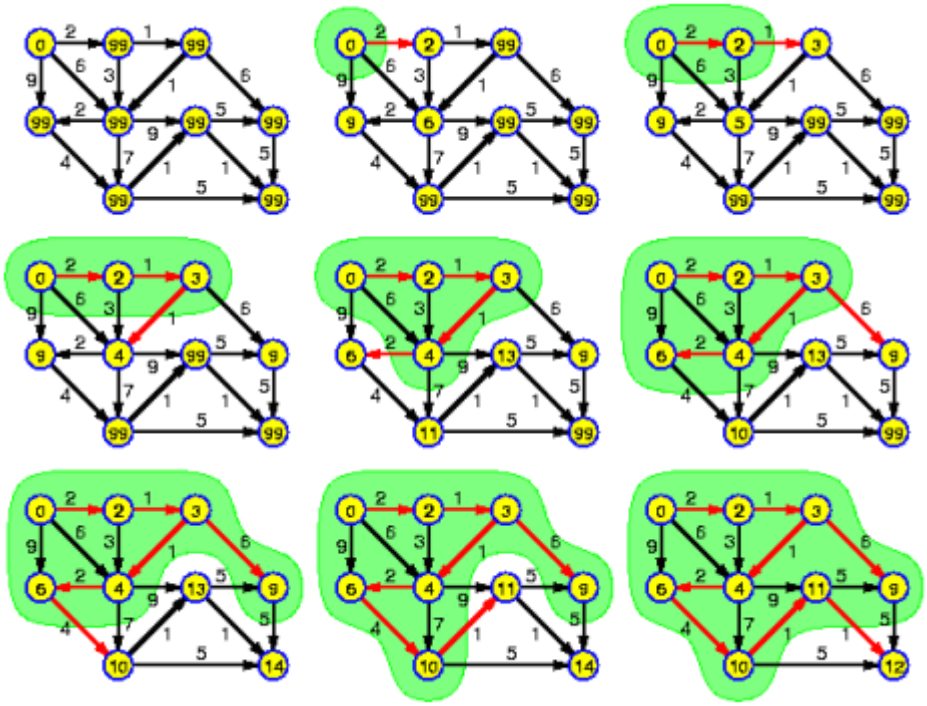

Figura 3 – Execução do algoritmo de Dijkstra (MARTINS, 2015)

3 METODOLOGIA PROPOSTA

Este Capítulo tratará sobre a metodologia e técnicas utilizadas no desenvolvimento deste trabalho. Usando como base o referencial teórico apresentado no Capítulo anterior.

3.1 ESCOLHA DA APRENDIZAGEM POR REFORÇO

Como mencionado no capítulo 2, o Algoritmo de Dijkstra poderia ser utilizado para a solução do problema proposta neste trabalho, no entanto, como abordado por Sichman (2021), atualmente atravessamos um período de euforia sobre os possíveis benefícios que a IA pode prover. Tal otimismo se justifica por uma conjunção de três fatores fundamentais: (i) o custo de processamento e de memória nunca foi tão barato; (ii) o surgimento de novos paradigmas, como as redes neurais profundas, possibilitados pelo primeiro fator e produzindo inegáveis avanços científicos; e (iii) uma quantidade de dados gigantesca disponível na internet em razão do grande uso de recursos tais como redes e mídias sociais. Sendo assim, foi escolhida a aprendizagem por reforço para a solução do problema abordado neste trabalho.

3.2 CRIAÇÃO DA MATRIZ

Um supermercado simulado está sendo representado por uma matriz 10x10, a mesma é criada utilizando a biblioteca NumPy em Python. Na Figura 4 é possível ver como o supermercado foi representado nessa matriz. Sendo que os números zeros (0) em branco, são células vazias e navegáveis pelo agente, os números uns (1) em azul, são células onde há gôndolas e possíveis objetivos do agente e os números dois (2) em vermelho, são representações dos obstáculos presentes, ou seja, onde o agente não pode percorrer.

0	0	0	0	0	0	2	0	0	0
0	0	0	0	0	0	0	0	0	0
0	1	0	1	0	1	0	1	0	0
0	0	0	0	0	0	0	0	0	0
0	1	0	0	2	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0
0	1	0	1	0	1	0	1	0	0
0	0	2	0	0	0	0	2	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Figura 4 – Representação da Matriz do Supermercado. Fonte: Autor

3.3 GERAÇÃO DA LISTA DE COMPRAS

Os itens da lista de compra foram selecionados pelo usuário através de uma janela aberta no momento da execução do programa, utilizando a biblioteca *Tkinter*[1] no Python. Como a Figura 5 mostra, a janela exibe os itens disponíveis para seleção, assim como as opções de adicionar ou remover itens da lista do usuário.

[1] https://docs.python.org/pt-br/3/library/tkinter.html

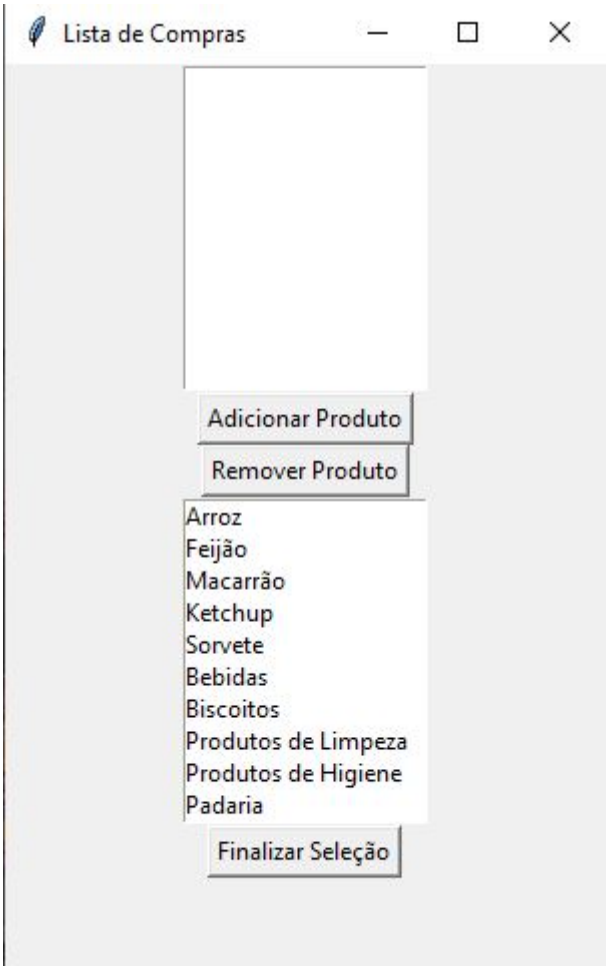

Figura 5 – Janela para Seleção de Itens da Lista. Fonte: Autor

3.3.1 Posição dos Obstáculos e Produtos

A figura 4 mostra a posição das gôndulas, obstáculos e espaços vazios do supermercado em questão. Todas essas posições foram criadas e inseridas pelo autor em um ambiente simulado e de teste. Este supermercado representado na matriz não foi baseado em um estabelecimento da vida real.

3.4 TREINAMENTO DO AGENTE

O agente está sendo treinado utilizando *Q-Learning* em um processo de 1000 episódios (1000 épocas), onde em cada episódio o agente pode percorrer, no máximo, 100 passos. Durante a execução do algoritmo, o agente percorre a matriz em busca de sinais de reforço (recompensas e punições) até encontrar o caminho onde ele obterá a maior recompensa até atingir seus objetivos.

3.5 AMBIENTE DE DESENVOLVIMENTO

Este trabalho fez uso de algumas tecnologias, bibliotecas e hardwares para o seu desenvolvimento. Abaixo serão abordados todos esses fatores.

3.5.1 Tecnologias Utilizadas

3.5.1.1 Bibliotecas Python

Devido aos motivos apresentados no referencial teórico deste trabalho, foi utilizada a linguagem Python para o desenvolvimento prático do mesmo. Foram utilizadas também, algumas bibliotecas que se mostraram essenciais para o êxito desta pesquisa. Uma delas foi a biblioteca NumPy[2], que de acordo com Alves et al. (2020), a melhor opção para se trabalhar com matrizes de uma forma segura é através das inúmeras bibliotecas e módulos disponíveis para esse fim. Dessa forma, destaca-se a biblioteca NumPy, que é uma biblioteca projetada com códigos de alta performance.

Além da biblioteca NumPy, também foram utilizadas outras duas bibliotecas no desenvolvimento deste trabalho, as bibliotecas *front-end Tkinter* e PyGame[3], para exibição da lista de compras e da melhor rota, respectivamente.

3.5.1.2 Visual Studio Code

O editor de código usado neste trabalho foi o Visual Studio Code[4]™da empresa Microsoft®. O Visual Studio Code é um editor de código-fonte leve, mas poderoso, que funciona no seu *desktop* e está disponível para Windows, macOS e Linux. Ele vem com suporte integrado para JavaScript[5], TypeScript[6] e Node.js[7] e tem um rico ecossistema de extensões para outras linguagens e ambientes de execução (como C++, Java, Python, PHP, Go, .NET) (MICROSOFT, 2023).

[2] https://numpy.org/doc/stable/
[3] https://www.pygame.org/docs/
[4] https://code.visualstudio.com/docs
[5] https://developer.mozilla.org/en-US/docs/Web/JavaScript
[6] https://www.typescriptlang.org/docs/
[7] https://nodejs.org/docs/latest/api/

3.5.2 Configuração de *Hardware*

Para elaborar e executar a aplicação por trás do trabalho, foi utilizado um computador pessoal com as seguintes configurações de *Hardware*:

Ambiente de Execução	
Arquitetura	x64_64
Processador	Intel®Core™i5 CPU @ 2.20 GHz
RAM	8 GB
Disco	1 TB

Tabela 1 – *Hardware* Utilizado. Fonte: Autor

4 RESULTADO

Após serem aplicadas as ferramentas e metodologias tratadas no Capitulo 3, foram obtidos os resultados a serem apresentados a seguir.

4.1 MODELO DE SAÍDA

Após o usuário selecionar os itens da lista dispostos na figura 5, e clicar em "Finalizar Seleção", o mesmo irá executar o algoritmo Q-Learning que, com base na disposição dos produtos no supermercado de acordo com a Figura 4, irá traçar a melhor rota de compras, tendo como *output* (saída) à matriz representada na Figura 6 usando a biblioteca PyGame. Abaixo são apresentados alguns resultados do modelo de saída obtidos após a execução do programa. Onde a célula em verde é o ponto de partida do agente, as células em vermelho são os objetivos do agente, as células em preto são os obstáculos e as células na cor azul são o próprio agente. Cada célula possui as iniciais "U", "D", "L"e "R"que significam, respectivamente, *UP* (cima), *DOWN* (baixo), *LEFT* (esquerda) e *RIGHT* (direita), acompanhadas pelo reforço obtido pelo agente para cada umas das ações.

A Figura 6 mostra cada um dos itens apresentados acima.

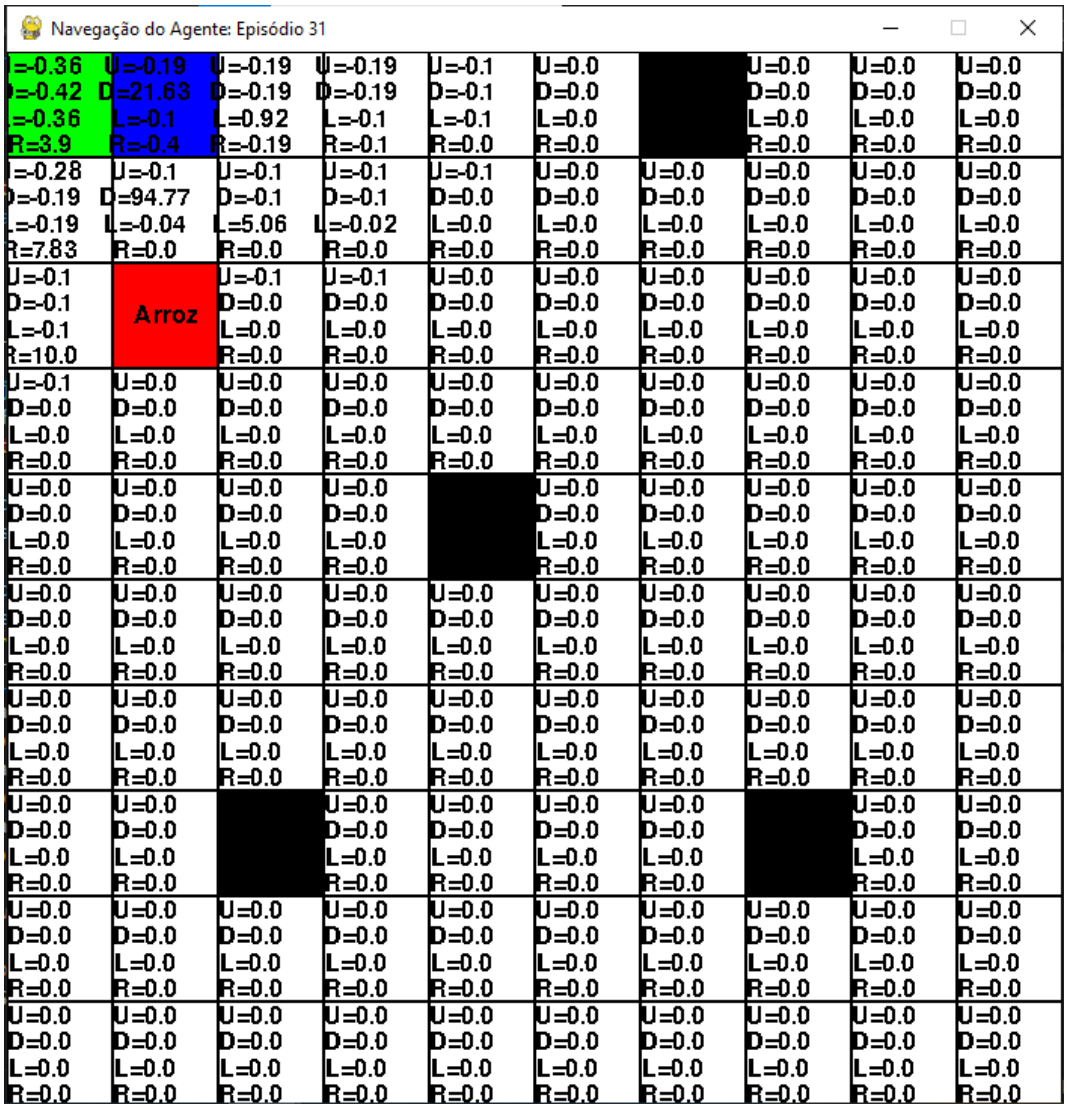

Figura 6 – Navegação do Agente com 1 objetivo. Fonte: Autor

A Figura 7 demonstra a matriz do supermercado representada na aplicação, seguindo o planejamento apresentado na Figura 4 no capítulo anterior.

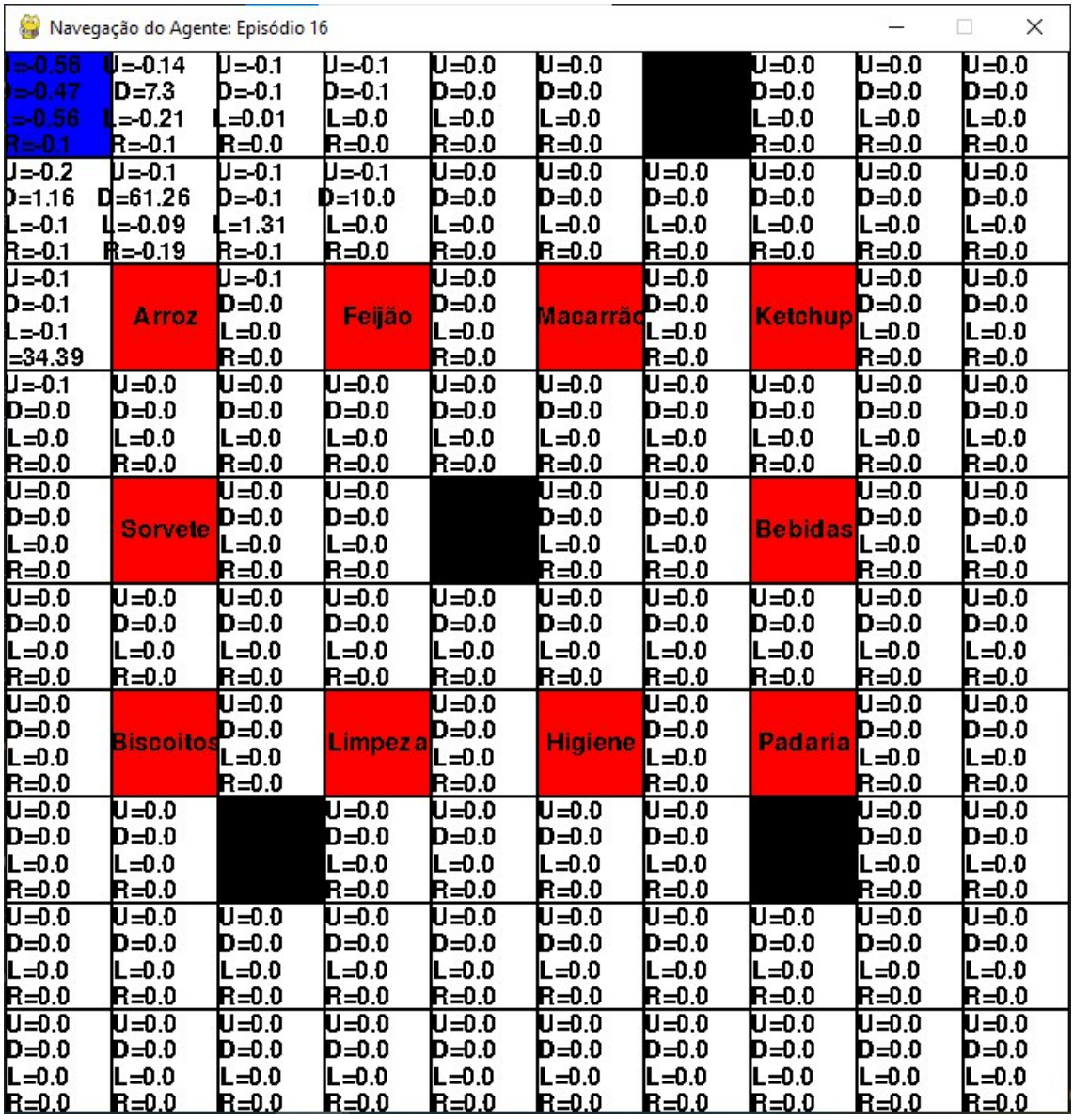

Figura 7 – Matriz do Supermercado na Aplicação. Fonte: Autor

A Figura 8 mostra o caminho, passo a passo, percorrido pelo agente do início (0, 0) até o seu primeiro objetivo e posteriormente para o seu segundo e terceiro objetivos.

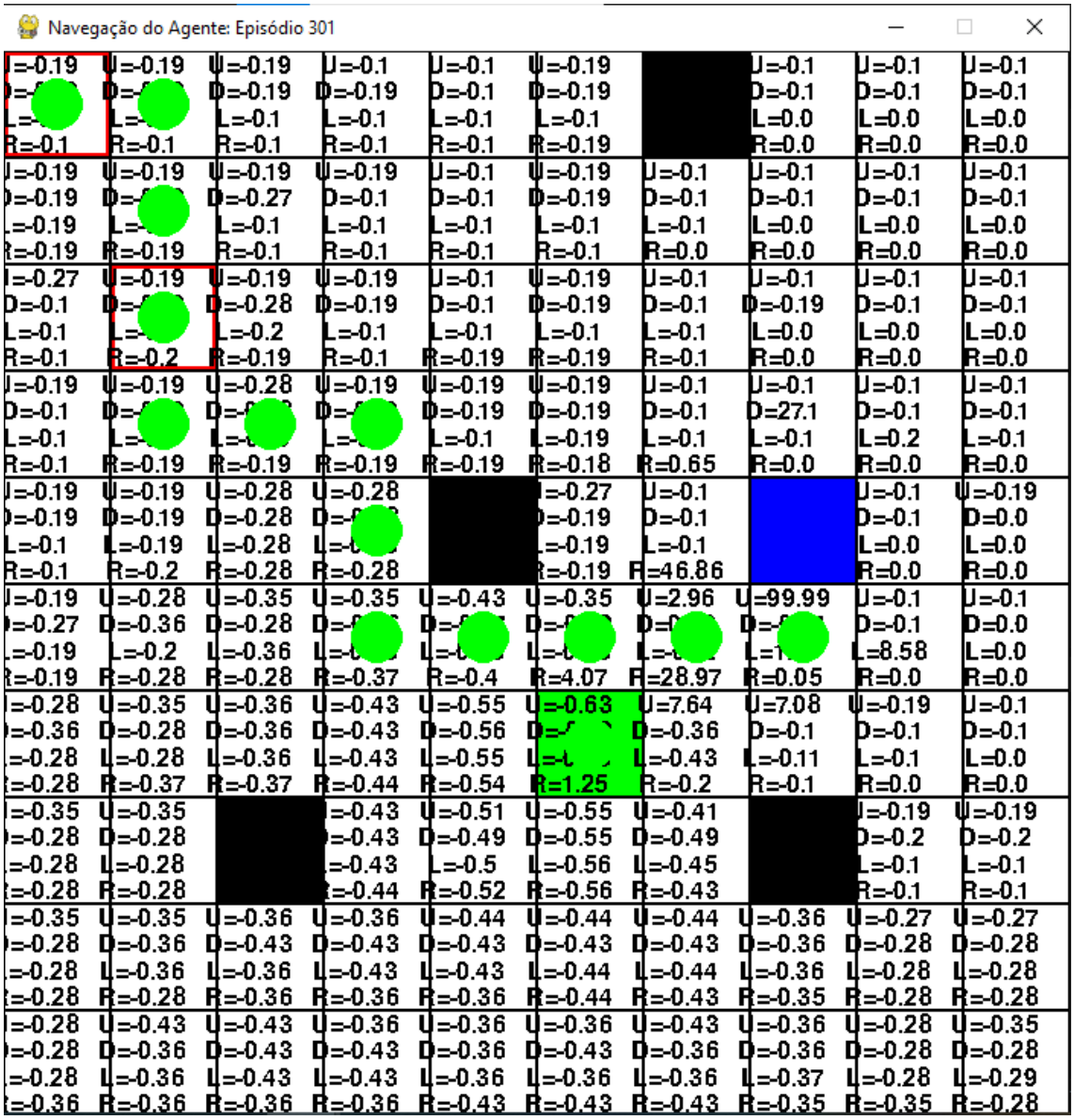

Figura 8 – Navegação do Agente com 3 objetivos. Fonte: Autor

Além das figuras apresentadas acima, o vídeo "Execução do Agente Q-Learning"[1] mostra a execução do algoritmo, onde o agente percorre toda a matriz calculando a melhor rota até atingir os seus objetivos.

[1] https://youtu.be/DZ6TdsugRUM

4.2 EFICIÊNCIA DO MODELO

O modelo se mostrou eficiente e retornou a melhor rota em todos os testes realizados.

A Figura 9 mostra que o agente percorreu o caminho com uma solução ótima para a quantidade total de objetivos disponíveis (10 produtos). O treinamento foi realizado com 1000 episódios. Sendo possível, no máximo, 100 passos por episódio.

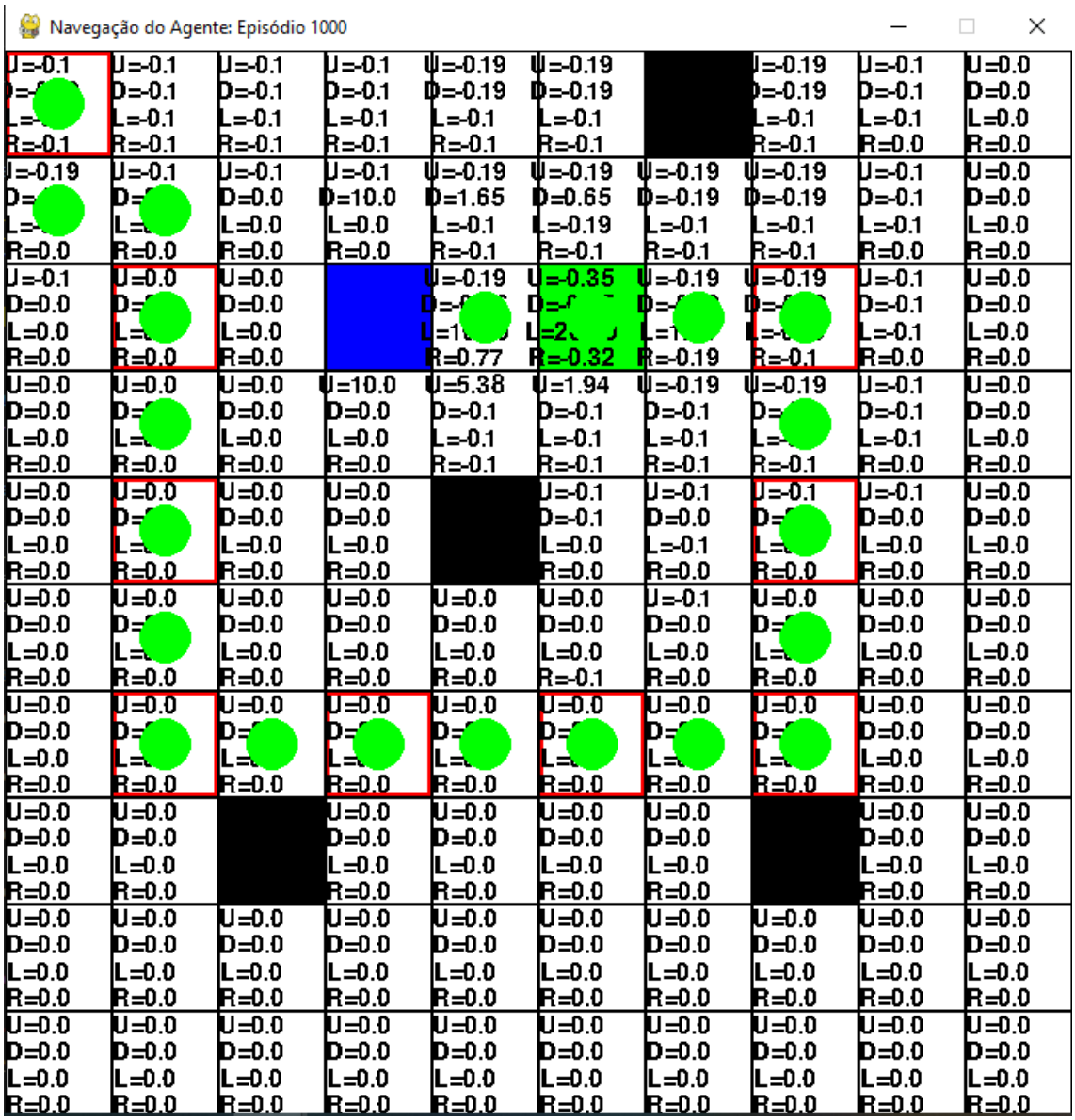

Figura 9 – Rota Ótima com 10 objetivos. Fonte: Autor

4.2.1 Gráficos de Convergência

A Figura 10 abaixo mostra o gráfico de convergência do algoritmo, gerado com o auxilio da biblioteca MatPlotLib[2] no Python. Onde, com o passar da execução dos episódio (épocas), o gráfico mostra que o agente diminui a sua punição recebida. Até que o agente encontra uma solução ótima para o problema e o gráfico pouco se altera.

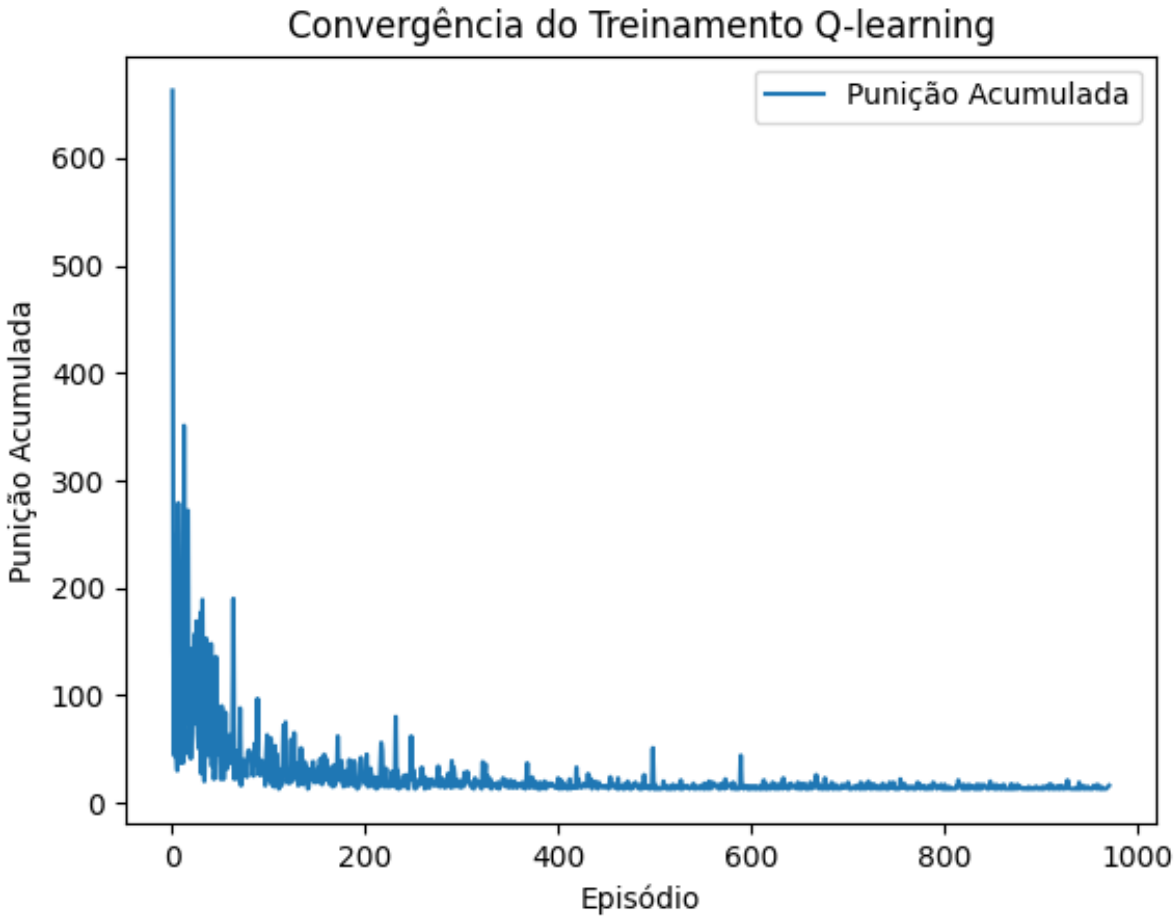

Figura 10 – Punição Acumulada. Fonte: Autor

Na Figura 11 é possível ver que, com o passar dos episódios, o agente converge para uma solução ótima e diminui a quantidade de passos dada em cada episódio. Além de que, com o passar do tempo, o agente diminui o seu caráter exploratório e passa a ser mais explotatório.

[2] https://matplotlib.org/stable/index.html

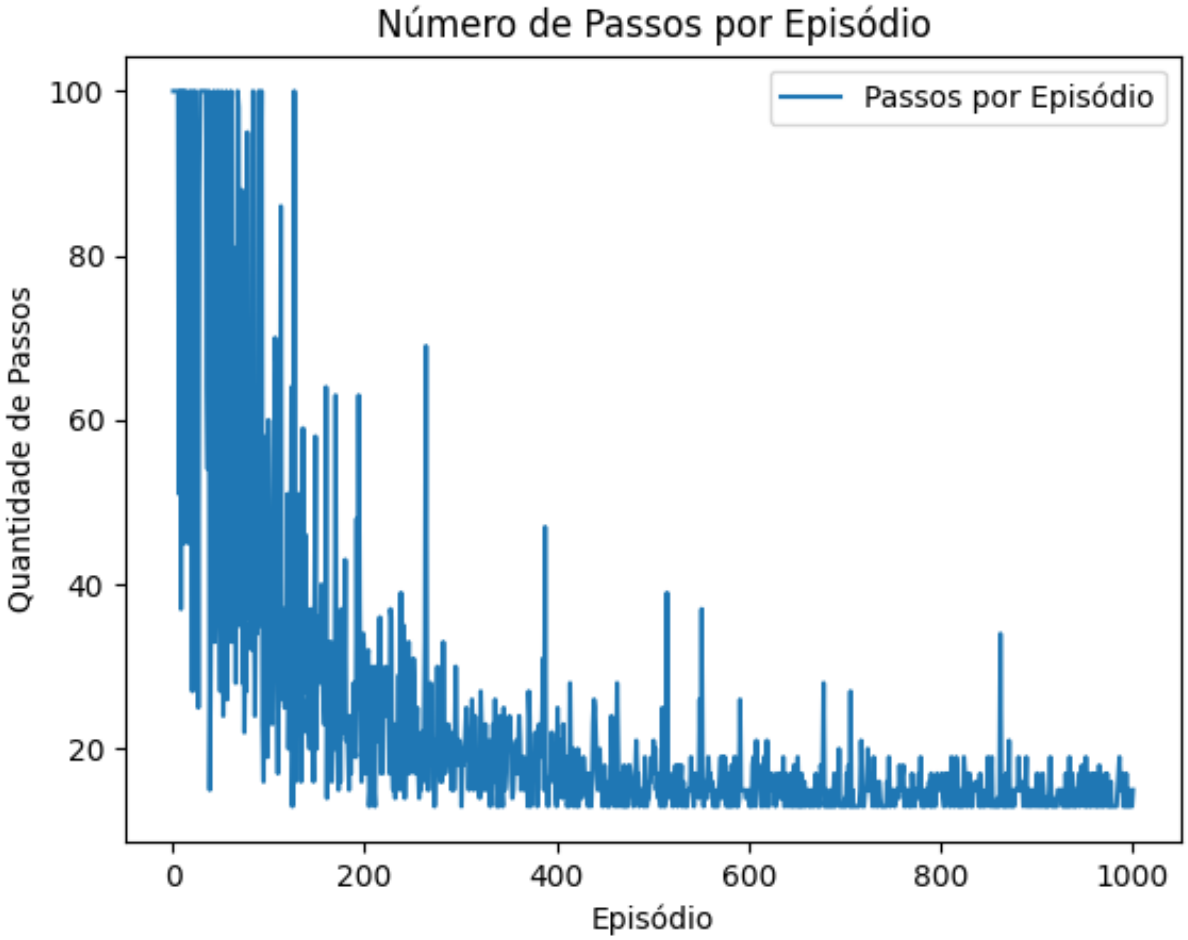

Figura 11 – Quantidade de Passos por Episódio. Fonte: Autor

4.2.2 Tempo de Execução

A execução do programa com as 1000 épocas, levou 627 segundos para ser concluída. As configurações da máquina que realizou os testes está disposta na Tabela 1, exibida no capítulo anterior.

5 CONSIDERAÇÕES FINAIS

Neste trabalho foi apresentado o problema, referencial teórico, metodologia e o resultado obtido. A solução proposta para o problema em questão foi a implementação de um algoritmo baseado em Aprendizagem por Reforço com o intuito de encontrar o caminho mais curto para atingir seus objetivos dentro de um ambiente de supermercado. Para tal, foi utilizada a fórmula do *Q-Learning* para calcular o reforço (recompensa ou punição), dado uma ação (*a*) em um estado (*s*).

Os objetivos deste trabalho foram atingidos de acordo com as expectativas iniciais e com o uso das técnicas apresentadas nos capítulos 2 e 3. É possível concluir que este trabalho possa ser utilizado por inúmeras pessoas, beneficiando não só os clientes como também os supermercados devido à oferecer a comodidade de os usuários do estabelecimento fazerem suas compras de maneira mais rápida e direta. Este trabalho também poderá ser aplicado à outras estruturas de supermercados e com diferentes disposições de produtos nas prateleiras, devido à fácil manutenção do código, alteração das posições dos produtos na matriz e alteração no tamanho da matriz.

Como sugestão para trabalhos futuros, podemos destacar:

a) Comparar a solução apresentada com outras disponíveis para o problema do caixeiro viajante, como o algoritmo de Dijkstra por exemplo. Apontando qual seria a melhor solução dentre elas.

b) Implementar o algoritmo em um robô físico para que ele guie os clientes ou então faça compras autônomas com base na lista pessoal dos clientes.

c) Uma possível integração com o supermercado para obter dados de compras dos clientes, como listas e frequências com que o cliente compra determinados produtos.

d) A possibilidade de os usuários alimentarem o sistema com as posições dos produtos

do supermercado, utilizando gamificação ou programa de recompensas como incentivo.

REFERÊNCIAS

ALVES, B. et al. *Python 101*. Cachoeira do Sul, RS, Brasil: Universidade Federal de Santa Maria, 2020.

BORGES, L. E. *Python para desenvolvedores: aborda Python 3.3*. [S.l.]: Novatec Editora, 2014.

CAMARGO, S. M.; TOALDO, A. M. M.; SOBRINHO, Z. A. O layout como ferramenta de marketing no varejo. In: *XXXIII Encontro da ANPAD*. [S.l.: s.n.], 2009.

CARVALHO, B. de. Algoritmo de dijkstra. *Universidade de Coimbra, Coimbra, Portugal*, 2008.

DENG, Y. et al. Fuzzy dijkstra algorithm for shortest path problem under uncertain environment. *Applied Soft Computing*, v. 12, n. 3, p. 1231–1237, 2012. ISSN 1568-4946. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1568494611004376>.

JúNIOR, E. P. F. D. *Aprendizado por reforço sobre o problema de revisitação de páginas web*. 73 p. Dissertação (Dissertação de Mestrado) — Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, Brasil, 2012. Inclui bibliografia.

LI, Y. Deep reinforcement learning: An overview. *arXiv preprint arXiv:1701.07274*, 2017.

LUDERMIR, T. B. Inteligência artificial e aprendizado de máquina: estado atual e tendências. *Estudos Avançados*, Instituto de Estudos Avançados da Universidade de São Paulo, v. 35, n. 101, p. 85–94, Jan 2021. ISSN 0103-4014. Disponível em: <https://doi.org/10.1590/s0103-4014.2021.35101.007>.

MARTINS, W. C. *Algoritmo para criação de rotas de compras econômicas*. 2015. Trabalho de Conclusão de Curso - Universidade de Brasília - UnB, Faculdade UnB Gama - FGA. 58 p. : il. (algumas color.) ; 30 cm. Orientador: Prof. Doutor Nilton Correia da Silva.

MICROSOFT. *Visual Studio Code Documentation*. 2023. Acessado em: 25 de novembro de 2023. Disponível em: <https://code.visualstudio.com/docs>.

MONARD, M. C.; BARANAUSKAS, J. A. Conceitos sobre aprendizado de máquina. *Sistemas inteligentes-Fundamentos e aplicações*, v. 1, n. 1, p. 32, 2003.

MORAES, A. C. d.; SILVA, J. d. O. A influência da atmosfera de varejo na satisfação e lealdade do consumidor. *Revista de Administração Contemporânea*, ANPAD, v. 21, n. 2, p. 153–172, 2017.

OLIVEIRA, L. F. d. J. *Identificação e representação automática de percursos de autocarros*. Dissertação (Mestrado) — Instituto Superior de Engenharia de Lisboa, 2015. Disponível em: <https://repositorio.ipl.pt/handle/10400.21/6009>.

RASCHKA, S.; PATTERSON, J.; NOLET, C. Machine learning in python: Main developments and technology trends in data science, machine learning, and artificial intelligence. *Information*, v. 11, n. 4, 2020. ISSN 2078-2489. Disponível em: <https://www.mdpi.com/2078-2489/11/4/193>.

SICHMAN, J. S. Inteligência artificial e sociedade: avanços e riscos. *Estudos Avançados*, Instituto de Estudos Avançados da Universidade de São Paulo, v. 35, n. 101, p. 37–50, Jan 2021. ISSN 0103-4014. Disponível em: <https://doi.org/10.1590/s0103-4014.2021.35101.004>.

SILVA, D. A.; ARROYO, C. S. Como o layout de um supermercado influencia a compra do consumidor? *Anais do 10º Congresso Nacional de Excelência em Gestão*, 2013.

SUTTON, R. S.; BARTO, A. G. *Reinforcement learning: An introduction*. [S.l.]: MIT press, 2018.

WATKINS, C. J.; DAYAN, P. Q-learning. *Machine learning*, Springer, v. 8, p. 279–292, 1992.