Universidade Federal Rural de Pernambuco
Departamento de Estatatística e Informática

Um Sistema de Recomendação para Auxílio à Reeducação Alimentar

Rodrigo Félix da Silva

Recife

Dezembro de 2015

Rodrigo Félix da Silva

Um Sistema de Recomendação para Auxílio à Reeducação Alimentar

Orientador: Cícero Garrozi

Coorientador: Rodrigo Gabriel Ferreira Soares

Monografia apresentada ao Curso de Bacharelado em Sistemas de Informação da Universidade Federal Rural de Pernambuco, como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

Recife

Dezembro de 2015

A Deus,
à minha família,
aos meus amigos

Agradecimentos

Agradeço primeiramente a Deus por ter me guiado e protegido durante todo este tempo.

Agradeço à minha família por terem acreditado em mim durante esta caminhada, em especial ao meu avô Manoel e avó Josefa.

Agradeço aos meus amigos que estiveram comigo, tantos os antigos como os novos, em especial a minha amiga Priscila.

Agradeço aos meus orientadores, pela disponibilidade em me orientar e pelas instruções passadas no decorrer deste trabalho.

Agradeço a todos os professores do BSI e a UFRPE, por terem uma grande parcela no conhecimento que adquiri durante o curso.

Resumo

O mercado de dispositivos móveis tem se expandido bastante nos últimos anos, principalmente pela grande oferta de modelos com preços mais acessíveis, que causaram uma crescente popularização dos smartphones, modelos que se tornaram cada vez mais indispensáveis. Além de utilizar os smartphones para comunicação, seja por ligações telefônicas ou por meio das redes sociais, as pessoas têm recorrido aos aplicativos móveis para auxiliá-las nas mais diversas tarefas do dia a dia, independente se estão em casa, no trabalho, na academia, no trânsito ou em qualquer outro lugar. O papel da tecnologia tem sido buscar nos trazer benefícios. Com os smartphones, um desses benefícios é a agilidade na busca por informações e gerenciamento de processos. Pesquisas sugerem que utilizar tecnologia móvel para intervenções na saúde influenciam o ganho de conhecimento e mudança de comportamento. Neste contexto, o objetivo deste trabalho foi desenvolver um aplicativo móvel para auxiliar o usuário a montar um plano de alimentação que esteja dentro de um limite estabelecido para algum nutriente, oferecendo sugestões de alimentos baseadas em suas preferências. Além disso, foram utilizados métodos de clusterização na base de dados para formar grupos de alimentos nutricionalmente similares, com o objetico de oferecer uma troca inteligente de alimentos no plano sugerido.

Palavras-chave: mineração de dados, nutrição, dieta alimentar, clusterização, métodos de agrupamento, sistema de recomendação, aplicativo móvel, android

Abstract

The market for mobile devices has expanded greatly in recent years, mainly by the wide range of more affordable models, causing a growing popularity of smartphones, which have become increasingly indispensable. In addition to using smartphones for communication, either through phone calls or through social networks, people have resorted to mobile applications to help them in various tasks of everyday life, irrespective whether they are at home, at work, at the gym, in traffic or anywhere else. The role of technology has seeked to bring us benefits. With smartphones, one of those benefits is the agility in the search for information and process management. Research suggests that use mobile technology for health interventions influence the knowledge and behavior change. In this context, the aim of this study was to develop a mobile application to help users prepare an eating plan within a limit for some nutrient, offering suggestions based on user preferences. In addition, clustering methods were used in the database to form clusters of nutritionally similar foods, aiming to offer a smart food exchange in the suggested plan.

Keywords: data mining, nutrition, diet, clustering methods, recommendation systems, mobile app, android

Sumário

1 Introdução **1**

1.1 Motivação 2

1.2 Objetivos 2

1.3 Organização do trabalho 3

2 Trabalhos relacionados **4**

2.1 Sistemas de recomendação de dietas 4

3 O problema **7**

3.1 Definição do problema 7

3.2 Pré-processamento 8

3.2.1 Seleção de dados 8

3.2.2 Limpeza de dados 11

3.2.3 Normalização dos dados 11

4 Mineração de dados, Técnicas de Agrupamento e Tecnologias **13**

4.1 Técnicas de Agrupamento 14

4.1.1 K-means 15

4.1.2 X-means 16

4.1.3 Expectation Maximization 19
4.1.4 Spectral Clustering 20
4.2 Validação de agrupamentos 23
4.2.1 O critério silhueta 23
4.3 Tecnologias utilizadas 24
4.3.1 Plataforma Android 24
4.3.2 Eclipse e Plugin ADT 27
4.3.3 SQLite 28
4.3.4 Weka 28
4.3.5 Biblioteca Colt 28

5 Sistema de recomendação alimentar proposto 29

5.1 Inserir alimento consumido 31
5.2 Relatório de consumo 31
5.3 Nutrientes nos alimentos 32
5.4 Cálculo da necessidade calórica diária 33
5.5 Sugestão de plano de alimentação 34
5.6 Troca de alimento no plano de alimentação 38

6 Considerações finais 42

6.1 Conclusão do trabalho 42
6.2 Trabalhos futuros 43

Lista de Tabelas

3.1 Nutrientes presentes na tabela de composição nutricional do IBGE [8] 10

5.1 Coeficiente de atividade física 34

5.2 Distribuição de calorias sugeridas para as refeições ao longo do dia. (adaptado) [2] 36

5.3 Experimento com Spectral Clustering - quantidade de clusters gerados 38

5.4 Experimento com Spectral Clustering - cálculo do Silhouette 39

Lista de Figuras

3.1 Tabela de composição nutricional dos alimentos consumidos no Brasil - IBGE [8]. 9

3.2 Histograma dos nutrientes 12

4.1 Execução do algoritmo K-means 16

4.2 Comportamento do K-means e X-means com clusters alongados 17

4.3 X-means: Improve-Params 18

4.4 X-means: Improve-Structure 18

4.5 X-means: passo 3 19

4.6 EM: atualização das probabilidades a cada iteração 20

4.7 Clusters distribuídos em forma que traria dificuldades para K-means e EM . 21

4.8 Passos do Spectral Clustering 22

4.9 Arquitetura da plataforma Android. 25

5.1 Diagrama de casos de uso 30

5.2 Tela inicial 30

5.3 Nova refeição 31

5.4 Relatório de nutrientes consumidos 32

5.5 Nutrientes do alimento 33

5.6 Cálculo da necessidade calórica diária 34

5.7 Sugestão de alimentação diária 35

5.8 Cálculo do Silhouette para os particionamentos encontrados utilizando K-means, X-means, EM e Spectral Clustering 39

5.9 Troca de alimentos da sugestão 41

Capítulo 1

Introdução

A vida moderna tem tirado grande parte do tempo livre das pessoas e é sempre tentador ceder às ofertas de comidas prontas e rápidas como *fast-foods*, a fim de se ganhar algum tempo em meio à rotina agitada. Cada vez mais, as pessoas não se preocupam com a alimentação e trocam um almoço com feijão e arroz por um hambúrguer com *milk-shake*, sem pensar nas futuras consequências dessa má alimentação.

E devido a esses hábitos alimentares pouco saudáveis, as pessoas têm desenvolvido doenças como obesidade, diabetes, hipertensão, dentre outras. Segundo a Organização Mundial da Saúde (OMS), 39% da população adulta mundial estava acima do peso em 2014 [34], o que representava um total de 1,9 bilhão de adultos. Já no Brasil, esse número sobe para alarmantes 52,5% da população, de acordo com pesquisa feita pelo Ministério da Saúde, Vigitel 2014 [44].

A principal causa são os alimentos consumidos terem uma baixa disponibilidade de nutrientes que são necessários para o bom funcionamento do organismo, ou ainda por terem disponibilidade em excesso. É crescente o número de pessoas que em sua dieta precisam restringir a quantidade consumida de algum nutriente (como sódio, carboidratos, proteínas, etc.), a fim de manter uma meta de consumo que seja adequada. Assim, gerenciar a dieta cada vez mais se torna uma parte de nossas vidas.

1.1 Motivação

A tecnologia tem estado cada vez mais presente na vida das pessoas, tanto para auxiliá-las nas tarefas do dia-a-dia como para trazer algum tipo de entretenimento. O segmento de celulares mostra uma grande prova disso, onde os aparelhos deixaram de ser apenas um meio de efetuar ligações e ganharam tanta importância que as pessoas passaram a organizar suas vidas por meio deles. Agendam compromissos, realizam anotações que anteriormente eram feitas em papeis, estudam, leem as notícias do dia, escutam música, realizam registros fotográficos, acessam contas bancárias, divertem-se com jogos e mais uma infinidade de opções que só aumentam a cada dia.

A área relacionada a dispositivos móveis está em constante crescimento, com novos modelos sendo lançados a cada dia. Associado a isso, também é crescente o número de aplicativos buscando espaço nesse mercado. Isto é demonstrado nos números das duas maiores lojas de aplicativos do mercado, a *App Store* da *Apple* e a *Google Play* do *Google*. A loja da *Apple* somava um total de 1,5 milhão de aplicativos disponíveis e 100 bilhões de downloads até o mês de junho deste ano, enquanto a loja do *Google* continha 1,6 milhão de aplicativos até a mesma data e 50 bilhões de downloads até julho de 2013, última divulgação deste dado [42].

Muitas categorias de aplicativos são dedicadas a prestar auxílio ao usuário em guardar e manipular informações sobre tarefas que ele realiza ao longo do dia, em momentos que não estão predefinidos, mas que ele gostaria de registrar assim que acontecessem. Nesse contexto estão em crescimento acelerado as áreas de Sistemas Sensíveis ao Contexto, Computação Ubíqua, Internet das Coisas *(IoT)*, entre outras. Como as pessoas estão próximas a seus celulares e tablets, há uma maior facilidade em acessar esses aplicativos a qualquer momento do dia, se comparado a aplicativos para desktops e notebooks, onde nem sempre é possível estar com eles. Também são exemplos os aplicativos nas categorias Finanças, Transporte e Saúde.

1.2 Objetivos

O principal objetivo deste trabalho é desenvolver um aplicativo móvel para a plataforma Android, com a tarefa de sugerir um plano de alimentação, em que os nutrientes dos alimentos se enquadrem dentro de um limite estabelecido pelo usuário.

Para atingir esse objetivo, o aplicativo também deverá:

- Permitir que o usuário possa informar os alimentos que foram consumidos e avaliá-los, para que os alimentos que mais forem adequados ao seu gosto e dieta tenham mais probabilidade de serem indicados em futuras sugestões.
- Prover um meio inteligente de sugestão de novos alimentos para uma refeição, quando o alimento anteriormente sugerido não for palatável ao usuário. Os novos alimentos sugeridos devem ter valores nutricionais compatíveis que não comprometam a dieta.
- Disponibilizar um relatório que informe a quantidade de nutrientes que foram consumidos em um dado período, para que o usuário possa acompanhar se suas metas de consumo estão sendo seguidas.

1.3 Organização do trabalho

O presente trabalho está organizado em 6 capítulos, descritos abaixo:

- No Capítulo 1 encontra-se a introdução;
- O Capítulo 2 apresenta alguns trabalhos relacionados a sistemas de recomendação alimentar;
- O Capítulo 3 faz a descrição do problema e mostra como foi feita a escolha e preparação da base de dados para a etapa de mineração;
- O Capítulo 4 descreve as características dos métodos de agrupamento e tecnologias utilizadas neste trabalho;
- O Capítulo 5 apresenta o sistema de recomendação alimentar proposto neste trabalho e os experimentos realizados com os métodos de agrupamento;
- No Capítulo 6 encontra-se as considerações finais;

Capítulo 2

Trabalhos relacionados

Há diversos aplicativos e sistemas dedicados a auxiliar o usuário no controle da dieta. A maior parte deles apenas fazem o registro dos alimentos que foram consumidos, para depois disponibilizar a quantidade de nutrientes presentes nos alimentos informados. Então cabe ao usuário verificar o andamento da sua meta para algum nutriente e manter-se alerta em não ultrapassá-la. Existem também os sistemas de recomendação de dietas que têm como fundamento propor refeições que já estejam adequadas aos anseios nutricionais do usuário, para que ele preocupe-se apenas em consumir o que foi sugerido.

Neste Capítulo, são apresentados trabalhos que abordam sistemas de recomendação relacionados a alimentação.

2.1 Sistemas de recomendação de dietas

- *DietSysNet* [14] é um sistema para recomendação de dietas que tem foco em auxiliar os nutricionistas a decidir um programa alimentar para os pacientes, buscando apresentar recomendações de prescrições através da técnica de Raciocínio Baseado em Casos (RBC) para que o nutricionista as avalie e tome a melhor decisão. RBC é uma técnica que utiliza o aprendizado ao analisar soluções de problemas passados para solucionar novos problemas.

  Foi dividido em dois módulos, um de uso exclusivo do nutricionista e outro do paciente. No módulo do nutricionista, ele pode efetuar o cadastro de seus pacientes, inserindo

suas informações básicas e suas respostas em relação às características, que são respostas a perguntas como *"frequência de consumo de embutidos"* ou *"tem diabetes?"*, formando assim o perfil de cada paciente. No módulo do paciente, ele pode ter acesso as suas prescrições, dados nutricionais dos alimentos e informar a evolução do seu quadro de saúde.

O sistema foi desenvolvido para ser disponibilizado através de página *web*, o que pode trazer a desvantagem que o usuário necessite de uma conexão com a internet sempre que precisar buscar uma informação no sistema, além de não ter uma boa usabilidade para acesso via celular, que seria melhor por meio de um aplicativo. Outra desvantagem é o usuário não poder informar algum alimento que ele comeu que não estava dentro da dieta, para que o nutricionista possa readequar sua prescrição com base nessa informação.

- *FooDroid* [40] é um sistema de recomendação de menu, feito para as cantinas da Universidade de Zurique e disponível como um aplicativo para Android. Ele tenta resolver o problema do usuário precisar navegar por várias listas de pratos até que consiga achar uma opção do seu gosto ao olhar o cardápio de um restaurante. Ele busca fazer recomendações com base nas preferências do usuário em relação a ingredientes e leva em consideração as avaliações que os pratos receberam de outros usuários.

  Uma desvantagem desse sistema é que ele recomenda pratos prontos, sem dar a opção que o usuário possa trocar um alimento do prato por outro, além da diversidade desses pratos ser bastante restrita, pois a base de dados são apenas os pratos disponíveis nas cantinas cadastradas.

- *Foodtracker* [18] é um aplicativo desenvolvido para Android que sua ideia é recomendar os restaurantes mais próximos que possuem o alimento que o usuário escolheu. Os alimentos são agrupados pela chamada *Dieta do semáforo*, onde eles são distribuídos em três grupos conforme a recomendação de consumo, sendo os grupos rotulados pelas cores verde (livre), amarela (com moderação) e vermelha (consumir excepcionalmente).

  Para que o aplicativo seja eficaz, ele necessita que se mantenha um levantamento atualizado dos alimentos que estão disponíveis nos restaurantes da região do usuário, o que vai se tornando difícil de manter à medida que cresce o número de cidades que o aplicativo deseja atender.

O aplicativo desenvolvido neste trabalho, além dos objetivos descritos na Seção 1.2, busca

trazer soluções para a maioria das desvantagens dos sistemas de recomendação acima mencionados.

Capítulo 3

O problema

3.1 Definição do problema

A maioria dos aplicativos destinados ao controle da dieta se baseiam no pós-consumo de alimentos, onde o usuário informa os alimentos que ele ingeriu e verifica o andamento do consumo total de algum nutriente em relação a uma meta preestabelecida. Geralmente utilizam a *Dieta dos pontos* [24], que lida com o consumo de calorias, onde cada alimento recebe uma pontuação no valor de um número inteiro e que é diretamente proporcional a quantidade de calorias que ele possui. O objetivo dessa conversão é que números inteiros são visualmente melhores para o usuário assimilar ao andamento de suas metas e mais fácil de manter o foco em uma quantidade de pontos que ele ainda tem disponível para consumo, como também os pontos que ele perdeu para refeições futuras caso tenha consumido um alimento com uma pontuação muito grande.

O problema dessa abordagem é que o usuário precisa ver as informações de cada alimento antes do consumo, caso esteja preocupado em não passar de certo limite para algum nutriente. Com o passar do tempo, esse processo repetitivo pode deixá-lo entediado, o levando a consumir alimentos sem análise prévia e resultando em exceder o limite de sua meta, sofrendo penalizações em seus pontos para refeições seguintes e por fim o induzir a desistência da dieta, por achar que esse procedimento não está atendendo as suas expectativas.

O foco do aplicativo deste trabalho é que o usuário informe uma meta de consumo para algum nutriente e o aplicativo monte um plano de alimentação para um dia, formando as

refeições, buscando alimentos na base de dados sem ultrapassar o limite informado, com alimentos que sejam de seu gosto e também permitindo a troca dos alimentos do plano sem que os novos alimentos sugeridos comprometam a meta da dieta.

Para isso, foi preciso encontrar tabelas de composição nutricional de alimentos, para seus dados serem extraídos e tratados, com a finalidade de serem usados na base de dados do aplicativo como fonte para as sugestões do plano de alimentação.

3.2 Pré-processamento

É uma importante etapa do processo de extração de conhecimento em base de dados, onde é feito a captação e tratamento dos dados, antes de serem usados na etapa de Mineração de Dados.

3.2.1 Seleção de dados

Na busca por tabelas de composição nutricional de alimentos, a maioria das tabelas encontradas estava disponível apenas em versões impressas de livros ou em formatos PDF. A *Tabela Sonia Tucunduva Philippi* [37] da nutricionista que dá nome a tabela, está disponível em livro e formato PDF, enquanto somente em livros temos a *Tabela de Equivalentes, Medidas Caseiras e Composição Química dos Alimentos* [35] e a *Tabela de Composição Química dos Alimentos* [21]. Esse formato as tornavam inviáveis de serem utilizadas, visto que, no primeiro caso, a leitura desses dados só poderia ser feitas manualmente, pois a utilização de um leitor OCR poderia resultar em algum dado identificado incorretamente, causando inconsistência na base de dados, principalmente alterando substancialmente o resultado dos algoritmos de clusterização. Quanto as tabelas disponíveis em formato PDF, durante o processo de conversão do arquivo, haveria de se usar bastante trabalho manual para ajustar os dados mal posicionados por conta da disposição de *layout* dessas tabelas no PDF e também retirar a quantidade de informações sem relevância que seriam copiadas.

Diante deste cenário, optou-se por analisar as tabelas disponíveis em formato de planilha, em razão de que seria necessária apenas a transformação dos dados para o ambiente da mineração e também a confiança na consistência dos dados seria maior. Assim, a lista

limitou-se a duas opções, que foram a tabela *TACO (Tabela Brasileira de Composição de Alimentos)* com 597 alimentos, produzida pela Unicamp em 2011 [7] e a *Tabela de composição nutricional dos alimentos consumidos no Brasil* com 1971 alimentos, produzida pelo IBGE em 2008 [8]. Além de apresentar uma maior quantidade de alimentos, a tabela fornecida pelo IBGE continha informações sobre as porções específicas de cada alimento, assim como qual delas era a porção padrão, então essa foi a tabela escolhida. A Figura 3.1 mostra um trecho dessa tabela.

Esta tabela possui 41 atributos, descritos abaixo:

- o 1º atributo representa o código do alimento, definido como um número inteiro.
- o 2º atributo representa a descrição do alimento.
- o 3º atributo representa o código de preparação, definido como um número inteiro.
- o 4º atributo representa a descrição da preparação.
- do 5º ao 41º atributo são representados os nutrientes, suas unidades de medida e a quantidade presente em uma porção de 100 gramas do alimento. Estes atributos estão relacionados na Tabela 3.1.

PESQUISA DE ORÇAMENTOS FAMILIARES 2008-2009

TABELA DE COMPOSIÇÃO NUTRICIONAL DOS ALIMENTOS CONSUMIDOS NO BRASIL

CÓDIGO DO ALIMENTO	DESCRIÇÃO DO ALIMENTO	CÓDIGO DA PREPARAÇÃO	DESCRIÇÃO DA PREPARAÇÃO	ENERGIA (kcal)	PROTEÍNA (g)
6300101	ARROZ (POLIDO, PARBOILIZADO, AGULHA	99	NAO SE APLICA	135,62	2,50
6300201	ARROZ INTEGRAL	99	NAO SE APLICA	130,95	2,56
6300701	MILHO (EM GRAO)	1	CRU(A)	160,14	3,32
6300701	MILHO (EM GRAO)	2	COZIDO(A)	160,14	3,32
6300701	MILHO (EM GRAO)	3	GRELHADO(A)/BRASA/CHURRASCO	160,14	3,32
6300701	MILHO (EM GRAO)	4	ASSADO(A)	160,14	3,32
6300701	MILHO (EM GRAO)	7	REFOGADO(A)	160,14	3,32
6300701	MILHO (EM GRAO)	13	ENSOPADO	160,14	3,32
6300701	MILHO (EM GRAO)	99	NAO SE APLICA	160,14	3,32
6300706	CANJIQUINHA DE MILHO EM GRAO	1	CRU(A)	79,68	1,24
6300706	CANJIQUINHA DE MILHO EM GRAO	2	COZIDO(A)	62,95	1,24
6300706	CANJIQUINHA DE MILHO EM GRAO	13	ENSOPADO	79,68	1,24
6300706	CANJIQUINHA DE MILHO EM GRAO	14	MINGAU	62,95	1,24
6300706	CANJIQUINHA DE MILHO EM GRAO	15	SOPA	31,44	0,95
6300706	CANJIQUINHA DE MILHO EM GRAO	99	NAO SE APLICA	62,95	1,24
6300707	XEREM DE MILHO	99	NAO SE APLICA	62,95	1,24
6301001	AMENDOIM (EM GRAO) (IN NATURA)	99	NAO SE APLICA	567,00	25,80
6301101	ERVILHA EM GRAO	99	NAO SE APLICA	109,09	5,36
6301201	FAVA (EM GRAO)	99	NAO SE APLICA	85,62	4,80

Figura 3.1: Tabela de composição nutricional dos alimentos consumidos no Brasil - IBGE [8].

Nome do nutriente	Unidade de medida
Energia	kcal
Proteína	g
Lipídeos totais	g
Carboidrato	g
Fibra alimentar total	g
Cálcio	mg
Magnésio	mg
Manganês	mg
Fósforo	mg
Ferro	mg
Sódio	mg
Sódio de adição	mg
Potássio	mg
Cobre	mg
Zinco	mg
Selênio	mcg
Retinol	mcg
Vitamina A (Equivalente de atividade de Retinol)	mcg
Tiamina (Vitamina B1)	mg
Riboflavina (Vitamina B2)	mg
Niacina (Vitamina B3)	mg
Equivalente de Niacina (Vitamina B3)	mg
Piridoxina (Vitamina B6)	mg
Cobalamina (Vitamina B12)	mcg
Folato (equivalente dietético de Folato)	mcg
Vitamina D (Calciferol)	mcg
Vitamina E (total de Alpha-Tocopherol)	mg
Vitamina C	mg
Colesterol	mg
Ácidos graxos saturados	g
Ácidos graxos monoinsaturados	g
Ácidos graxos poliinsaturados	g
Ácido graxo poliinsaturado 18:2 (Linoléico)	g
Ácido graxo poliinsaturado 18:3 (Linolênico)	g
Ácidos graxos trans total	g
Açúcar total	g
Açúcar de adição	g

Tabela 3.1: Nutrientes presentes na tabela de composição nutricional do IBGE [8]

3.2.2 Limpeza de dados

Nesta etapa, valores como * ou – quando algum alimento não apresentava certo nutriente foram substituídos por 0. Também foram retirados caracteres especiais que causariam problemas na leitura dos dados pelos algoritmos na fase de mineração, como aspas simples e duplas.

3.2.3 Normalização dos dados

Nesta etapa foi ajustada a escala dos valores que representavam a quantidade dos nutrientes presentes nos alimentos, para que nutrientes com escala maiores não tivessem influência superior aos outros durante o processo de mineração. Para a nova escala, optou-se por deixar os valores no intervalo entre 0 e 1, aplicando a Equação (3.1):

$$V' = \frac{(V - Min)}{(Max - Min)} \tag{3.1}$$

onde:

- V' é o novo valor para o nutriente, agora normalizado;
- V é o valor do nutriente antes da normalização;
- Min é o valor mínimo do nutriente presente na base a ser normalizado;
- Max é o valor máximo do nutriente presente na base a ser normalizado.

Finalizando a etapa de pré-processamento, foram criados *scripts* na linguagem *python*, para exportar os dados da planilha e reestruturá-los no formato de arquivo ARFF, que é o formato aceito pelo programa *Weka*, responsável pela tarefa de mineração, detalhada no Capítulo 4. Baseado em outros trabalhos [13] [11] e nos nutrientes mais comumente presentes na informação nutricional em embalagem de alimentos, os nutrientes que foram selecionados para a continuidade do processo de extração do conhecimento, bem como sua presença no aplicativo, foram: Proteínas, Lipídeos, Carboidratos, Fibra Alimentar, Cálcio, Ferro, Sódio, Colesterol e Energia. A Figura 3.2 mostra o histograma desses nutrientes, após a normalização dos valores.

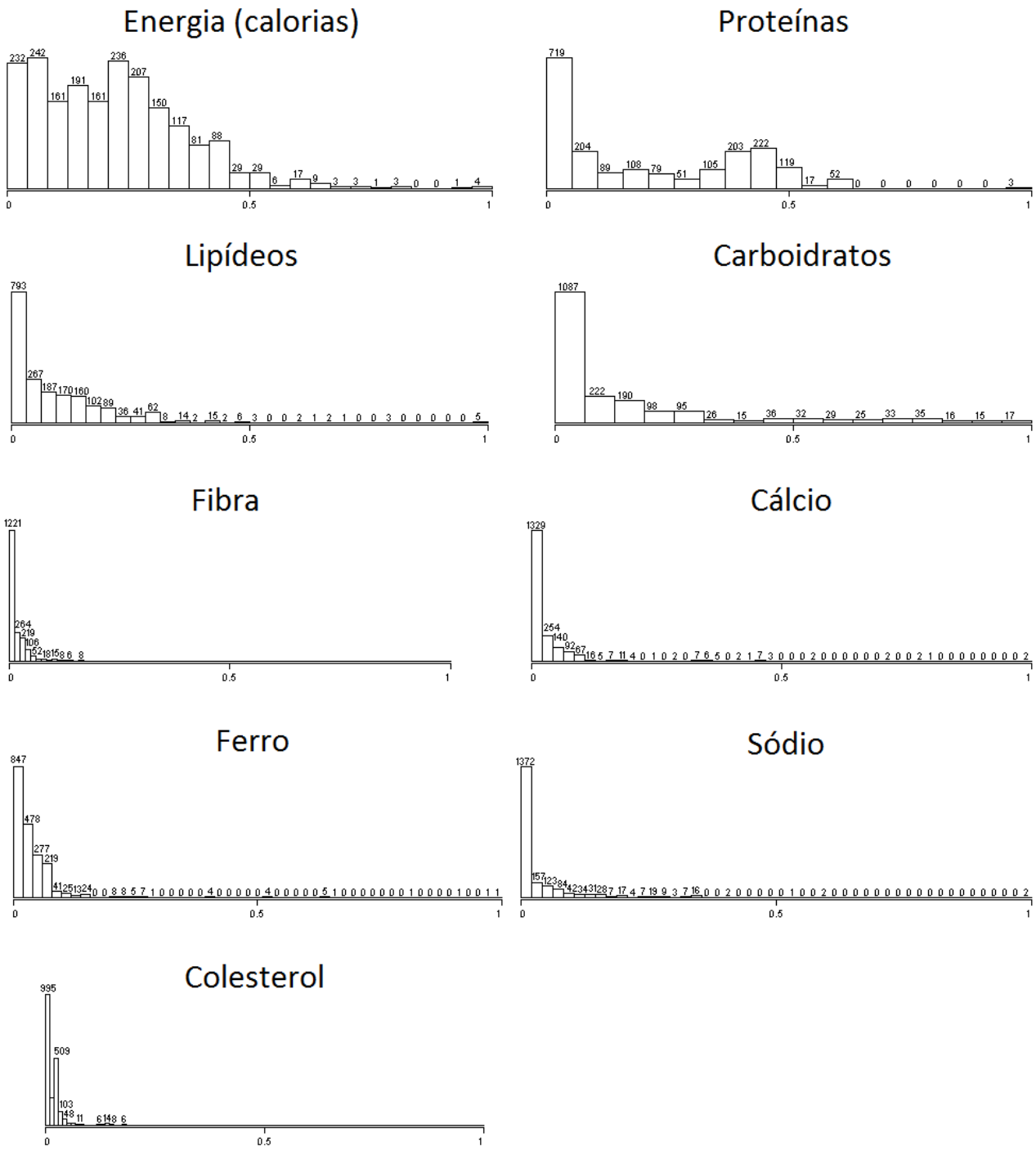


Figura 3.2: Histograma dos nutrientes

Capítulo 4

Mineração de dados, Técnicas de Agrupamento e Tecnologias

A mineração de dados é uma das etapas do processo de descoberta de conhecimento em base de dados. Do inglês *Data Mining*, as técnicas utilizadas nesse processo buscam descobrir tendências no conjunto de dados, que tenham algum significado, e que não poderiam ser encontradas por meios de exploração tradicionais, por estes padrões estarem ocultos, serem complexos ou alcançáveis apenas através de inferências.

Entre as principais tarefas da Mineração de Dados, as mais comuns são:

- Associação: busca detectar regras que indiquem um padrão de ocorrência simultânea de eventos [38]. Por exemplo, a descoberta de quais produtos são frequentemente comprados juntos em um supermercado. Outro exemplo seria, ao analisar uma base de dados de um hospital, detectar que pacientes que apresentam determinados sintomas tendem a ser diagnosticados com uma determinada doença.

- Sumarização: permite obter uma visão mais compacta da base de dados, identificando características sem que estas estejam influenciadas por valores que estão muito inconsistentes em relação aos demais valores do conjunto de dados [5].

- Classificação: está relacionada à identificação da classe que um determinado elemento pertence. Para isso, é necessária a presença de elementos que já estejam rotulados com as classes corretas que eles pertencem, com o propósito que esse conjunto seja

utilizado como base de treinamento, onde o algoritmo classificador poderá "aprender" suas características e classificar a qual categoria um novo elemento irá pertencer [38].

- Regressão: similar a Classificação, mas seu uso ocorre apenas com valores numéricos. É utilizada para estimar um valor para uma variável contínua desconhecida, por meio da análise das demais variáveis [16]. Um exemplo seria analisar uma base de dados contendo registros de vendas de imóveis de um determinado bairro e as características desses imóveis, como quantidade de quartos, dependência de empregada, quantidade de banheiros, etc. Analisando os valores das vendas e transformando as características do imóvel em variáveis, seria possível que um morador do bairro pudesse estimar um valor de venda pra seu imóvel por meio de Regressão.

- Detecção de anomalias: tem a tarefa de identificar comportamentos que fogem de um determinado padrão, como valores *outliers* ou ruídos. É uma tarefa bastante utilizada para detectar fraudes financeiras, como detectar um cliente efetuando várias compras em pouco espaço de tempo em lugares distantes de onde ele costuma usar seu cartão ou efetuando uma movimentação financeira que não condiz com seu histórico [38].

- Agrupamento: A clusterização ou agrupamento é uma técnica que consiste em particionar os elementos de uma base de dados em subconjuntos denominados *clusters*, onde a similaridade é o principal critério para o agrupamento, não sendo necessário que se defina classes antecipadamente como na Classificação [23].

Neste trabalho foram utilizadas técnicas de Agrupamento, com o objetivo de descobrir grupos de alimentos na base de dados que tenham valores nutricionais similares, utilizando para isso métodos não supervisionados. Estes grupos são utilizados no processo de troca de alimentos no plano de alimentação sugerido no aplicativo, detalhado no Capítulo 5. Os principais métodos de agrupamento serão detalhados na seção seguinte.

4.1 Técnicas de Agrupamento

Agrupamento ou clusterização faz parte de um paradigma chamado Aprendizado Não Supervisionado, onde o resultado que se busca ao analisar uma base de dados não é indicado por um agente externo, não há classes predefinidas e nem necessidade de conhecimento prévio sobre elas. Na clusterização, o algoritmo é auto-organizado e reconhece os padrões por si

próprio. Com esses padrões reconhecidos, os dados são organizados de modo que elementos com características mais próximas integrem o mesmo grupo, chamados de *cluster*, ao mesmo tempo em que tenham características distantes de integrantes de outros grupos.

Na literatura existem diversos métodos destinados a realizar esse processo, que apresentam seus próprios aspectos, sendo cada um fundamentado em uma definição de *cluster* [19]. As seções a seguir apresentam, brevemente, as características dos métodos que foram utilizados neste trabalho para realização de agrupamentos.

4.1.1 K-means

O *K-means* é o mais simples algoritmo de agrupamento existente e um dos mais comumente utilizados [29] [30]. Consiste em construir uma partição contendo k *clusters*, onde os elementos apresentem uma alta similaridade para outros elementos no mesmo grupo e baixa similaridade para elementos de grupos distintos. É preciso definir antecipadamente o número de *clusters* que serão formados na partição, representado pelo parâmetro k. Os passos gerais do algoritmo são:

- Passo 1. Selecione aleatoriamente k elementos, que inicialmente irão representar os centroides (centro de gravidade dos *clusters*).
- Passo 2. Atribua cada elemento remanescente ao *cluster* cujo centroide tenha a menor distância para o elemento.
- Passo 3. Para cada *cluster*, atualize a posição de seu centroide, que será determinada pela média dos elementos pertencentes ao *cluster*.
- Passo 4. Repita os passos 2 e 3 até que nenhum elemento seja realocado em um *cluster* diferente.

A Figura 4.1 demonstra os passos descritos acima considerando $k = 3$, onde: (a) elementos no estado inicial; (b) escolha dos centroides iniciais e atribuição dos elementos aos *clusters*; (c) atualização da posição dos centroides; (d) reatribuição dos elementos aos *clusters*; (e) formação dos *clusters* definitivos.

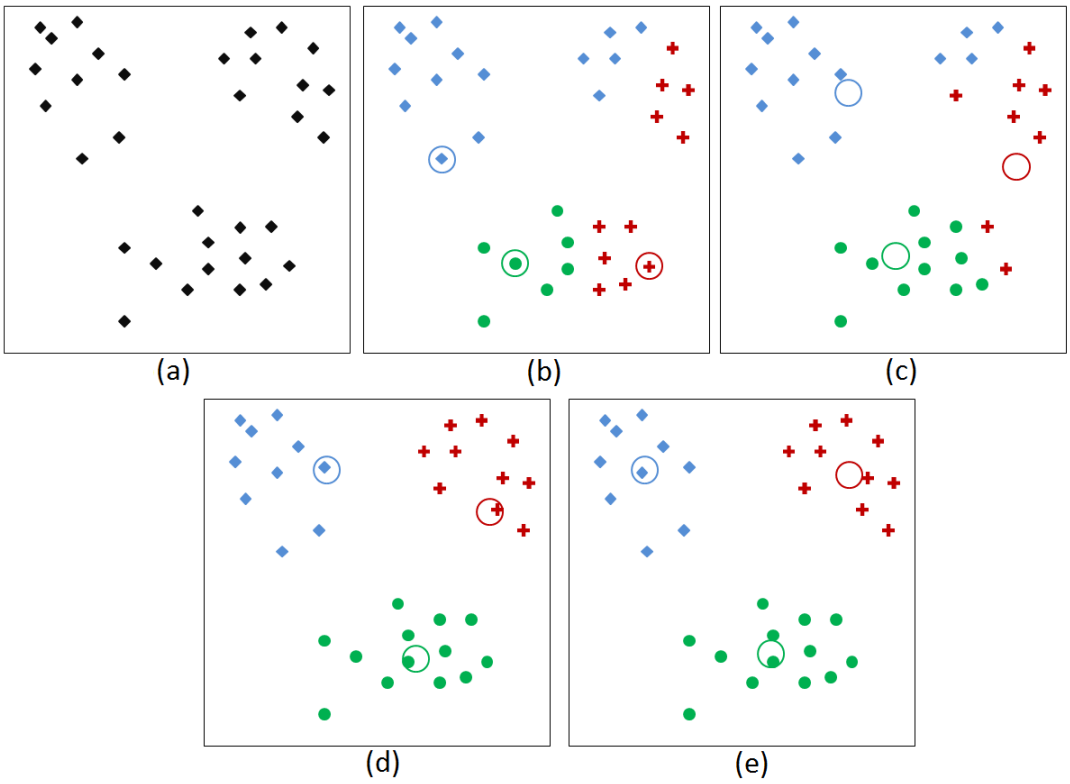


Figura 4.1: Execução do algoritmo K-means

A necessidade que o utilizador deva informar o valor de k (o número de *clusters*), pode ser considerado como uma desvantagem, visto que pode ser informado um valor que não corresponda a um k ótimo. Além disso, outros problemas apresentados pelo *K-means* são:

- Os dados são limitados a serem numéricos [26].
- É sensível a valores *outliers*. Um único objeto com valor muito extremo pode modificar consideravelmente a distribuição dos dados [25].
- Não é propício a descobrir *clusters* que não tenham forma convexa [4].

4.1.2 X-means

Proposto por D. Pelleg e A. Moore [36] como uma extensão do *K-means* para resolver três problemas encontrados nele:

- O parâmetro k ser fixo e previamente fornecido pelo usuário;

- A busca estar propensa a mínimos locais;
- Deficiente em termos de ser computacionalmente escalável.

O fato do *K-means* utilizar medidas de distância, como a Euclidiana, faz com que o algoritmo não encontre *clusters* que não tenham forma convexa (Figura 4.2 a), resultando que os agrupamentos encontrados apresentem forma esférica, misturando os dados de *clusters* não convexos (Figura 4.2 b). Então o *X-means* tenta minimizar esse problema usando uma quantidade maior de *clusters* (Figura 4.2 c). O intervalo de valores possíveis para k precisa ser definido por meio dos parâmetros k *mínimo* e k *máximo*.

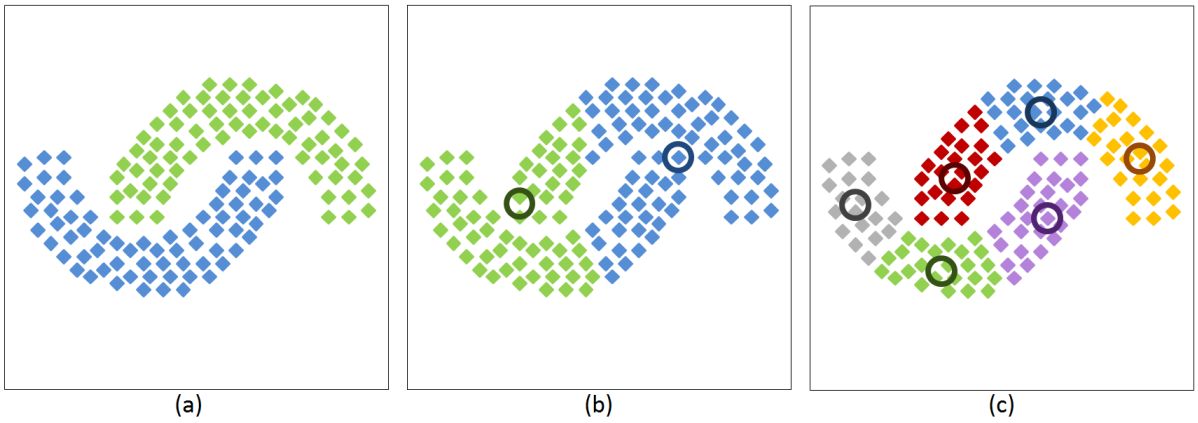


Figura 4.2: Comportamento do K-means e X-means com clusters alongados

O número de *clusters* ótimo é definido utilizando o Critério de Informação Bayesiano (BIC) ou o Critério de Informação de Akaike (AIC), que são medidas para escolha de modelos bastante utilizadas na literatura [43], que penalizam a verossimilhança, para que um modelo mais parcimonioso seja selecionado [17]. Cada *cluster* é recursivamente dividido em dois utilizando *K-means* e a iteração de divisão continua até ser terminada com base no valor estimado do BIC ou AIC, dentro dos limites fornecidos. Uma *kd-tree* é usada para armazenar os dados estatísticos e assim acelerar a velocidade da computação.

Os passos gerais do algoritmo são:

Passo 1: *Improve-Params* - neste passo aplica-se o algoritmo *K-means* para k *clusters* até convergir, onde k é igual ao k *mínimo* fornecido como valor de limite inferior. A Figura 4.3 demonstra o resultado final deste passo considerando $k = 3$.

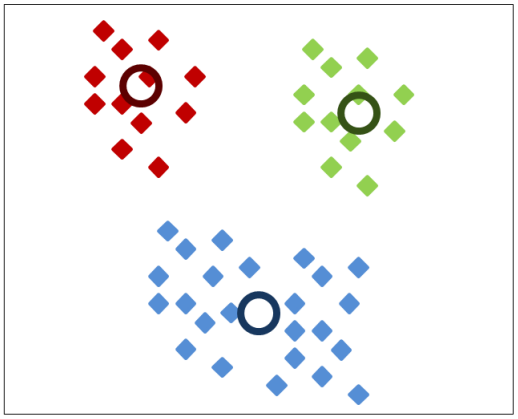


Figura 4.3: X-means: Improve-Params

Passo 2: *Improve-Structure* - a estrutura é melhorada dividindo o centro de cada *cluster* em dois filhos, dispostos em sentidos opostos em um vetor (Figura 4.4 a). Em cada um dos filhos, executa-se o *K-means* localmente (Figura 4.4 b). A decisão entre escolher o centro atual ou seus filhos é feita comparando os valores BIC das duas estruturas (Figura 4.4 c).

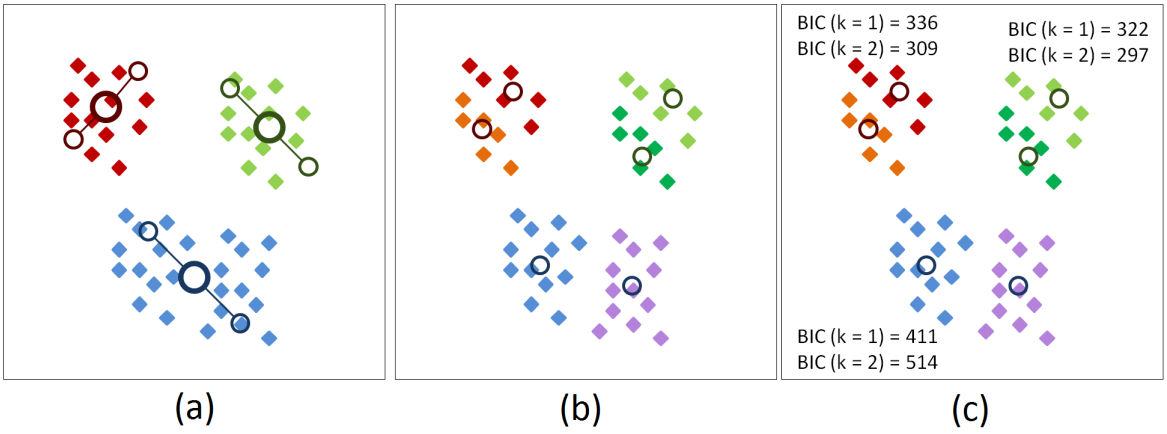


Figura 4.4: X-means: Improve-Structure

Passo 3: o terceiro passo indica a repetição dos passos 1 e 2 até que o valor atual de k ultrapasse o k *máximo* fornecido. Após isso, o algoritmo informa qual foi o modelo encontrado durante a busca que obteve a melhor pontuação (Figura 4.5).

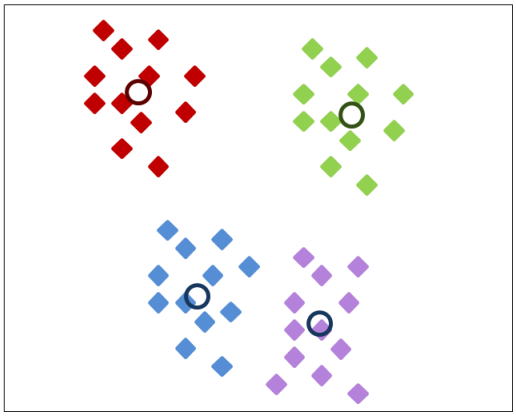


Figura 4.5: X-means: passo 3

4.1.3 Expectation Maximization

O Expectation Maximization (*EM*) [9] é uma variação do algoritmo *K-means*, que difere na maneira de associar cada elemento do conjunto de dados aos *clusters*. Enquanto o *K-means* atribui cada elemento a um único *cluster*, no *EM* cada elemento é associado a uma distribuição de probabilidades que ele pertença a cada um dos *clusters*, que são representados por Gaussianas, sendo o objetivo do algoritmo maximizar essas probabilidades.

O algoritmo é dividido em dois passos:

- Expectativa: Os elementos do conjunto de dados são associados aos *clusters*, de acordo com as probabilidades calculadas pela função modelo de cada *cluster*, com a atual estimativa dos parâmetros. Esses parâmetros podem ser iniciados aleatoriamente.
- Maximização: Nesta etapa é feita uma nova estimativa para os parâmetros da função modelo, levando em consideração os novos pesos dos elementos do conjunto, que foram avaliados no passo anterior, e então retorna ao primeiro passo até que se alcance a convergência.

Segundo [22], os passos de expectativa e maximização estão interligados, pois as novas probabilidades calculadas na fase de maximização serão utilizadas para realizar a inferência na fase da expectativa.

A Figura 4.6 demonstra a mudança de distribuição de probabilidades de atribuição aos *clusters* em cada elemento, ilustrada nas cores de cada elemento, após a iteração.

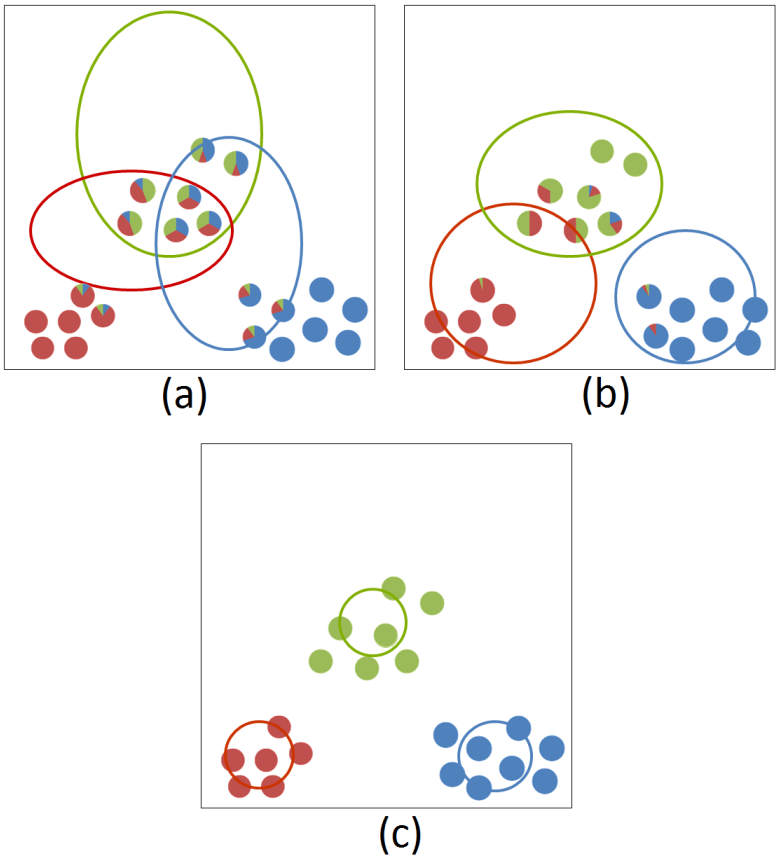


Figura 4.6: EM: atualização das probabilidades a cada iteração

4.1.4 Spectral Clustering

Surgido recentemente como alternativa promissora aos algoritmos tradicionais [33], a ideia fundamental do agrupamento espectral é agrupar pela afinidade, ou seja, o que será levado em consideração é a ligação que cada elemento tem com outro e não a sua localização absoluta. Dificilmente *K-means* ou *EM* fariam um grande trabalho caso procurassem por *clusters* em um conjunto de elementos que tivesse uma distribuição como a demonstrada na Figura 4.7, mas este não seria um grande problema para o *Spectral*, justificado pelo paradigma que ele adota para fazer os agrupamentos.

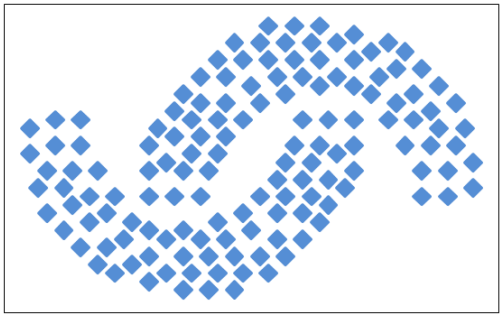


Figura 4.7: Clusters distribuídos em forma que traria dificuldades para K-means e EM

O principal artifício do algoritmo é transformar as representações dos elementos em características, que irão definir as propriedades dos *clusters*, e então algoritmos tradicionais como o *K-means* podem ser usados para fazer o agrupamento. Podemos dizer que o agrupamento espectral funciona como um pré-processamento para outros algoritmos.

Os passos gerais do algoritmo são:

Passo 1: Crie a matriz de afinidade $A \in \mathbb{R}^{n \times n}$. Para isso, assumindo que cada ponto de dados é representado por um vértice em um grafo, tal que o conjunto de vértices $V = \{v_1, ..., v_n\}$, tomando-se os vértices v_i e v_j e aplicando-se uma função de similaridade que irá definir o valor de A_{ij}. Caso o valor retornado pela função seja maior que um limiar definido, então esses vértices estão ligados por uma aresta. Na implementação do algoritmo sugerida por Ng et al., 2001 [33], a função de similaridade é definida pela Equação (4.1), onde $S = \{s_1, ... \ s_n\}$ é o conjunto de pontos de dados e σ^2 controla o quanto a afinidade A_{ij} cai de acordo com a distância entre s_i e s_j.

$$A_{ij} = exp(- \parallel s_i - s_j \parallel^2 /2\sigma^2) \text{ if } i \neq j, \text{ and } A_{ii} = 0 \tag{4.1}$$

Passo 2: Calcule a matriz Laplaciana L. É ressaltado por Nascimento, 2010 [32] que matrizes Laplacianas são de grande importância para o agrupamento espectral. As diferentes implementações dos algoritmos adotam diferentes variantes de Laplacianas, comumente se são normalizadas ou não-normalizadas. Implementações populares como Ng et al., 2001 [33] e Shi and Malik, 2000 [41] adotam a matriz Laplaciana normalizada nas variantes L_{sym} (matriz simétrica) e L_{rw} (caminhada aleatória) respectivamente. A Equação (4.2) define L_{sym}, enquanto L_{rw} está definida na Equação (4.3), onde L é a matriz Laplaciana (não-normalizada), A a matriz de adjacência, D a matriz de graus e I a matriz identidade.

$$L_{sym} = D^{-1/2} L D^{-1/2} = I - D^{-1/2} A D^{-1/2} \tag{4.2}$$

$$L_{rw} = D^{-1}L = I - D^{-1}A \tag{4.3}$$

A matriz Laplaciana não-normalizada é definida em Equação (4.4), onde A é a matriz de adjacência e D a matriz de graus.

$$L = D - A \tag{4.4}$$

Passo 3: Encontre os k autovetores $\{x_1, x_2, ..., x_k\}$ de L com os maiores ou menores autovalores, onde esta escolha vai estar relacionada a qual matriz Laplaciana foi utilizada.

Passo 4: Forme a matriz $X \in \mathbb{R}^{n \times k}$ com os vetores $\{x_1, x_2, ..., x_k\}$ como colunas.

Passo 5: Para cada linha i de X, seja $y_i \in \mathbb{R}^k$ relativo a i-ésima linha.

Passo 6: Forme *clusters* nos pontos $(y_i)_{i=1,...,n}$ utilizando *K-means*.

A Figura 4.8 mostra um resumo dos passos acima. Dado um conjunto de elementos (a). Cria-se a matriz de afinidade dos elementos (b), busca-se os autovetores com maiores autovalores (c) e os remapeia em um novo espaço, que é clusterizado facilmente com algoritmos de agrupamento tradicionais como *K-means* (d).

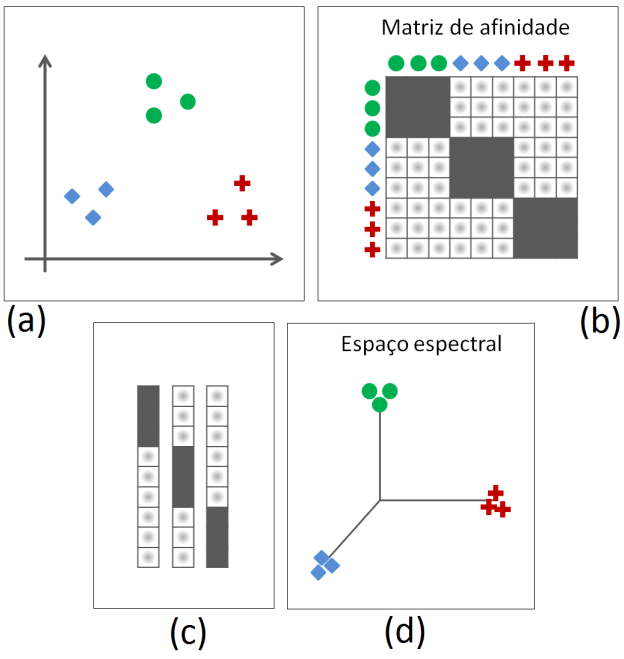


Figura 4.8: Passos do Spectral Clustering

O agrupamento espectral possui muitas vantagens se comparado a outros algoritmos tradicionais. Segundo Huiqing e Junjie [45], tem uma implementação simples e pode ser resolvido de maneira eficiente utilizando métodos de álgebra linear, além de conseguir melhores resultados que os algoritmos clássicos.

4.2 Validação de agrupamentos

Na clusterização de dados, um grande desafio é a tarefa de avaliar o quanto um algoritmo foi preciso em formar um particionamento com uma alta similaridade *intra-cluster* e a menor similaridade possível *extra-cluster*. A literatura nos traz diversos métodos para avaliar a consistência desses *clusters*, sendo *CH* (Calinski and Harabasz, 1974 [3]), *DB* (Davies and Bouldin, 1979 [6]), *Dunn* (Dunn, 1974 [10]) e *Silhouette* (Rousseeuw, 1987 [39]) métodos bastante populares. Todos os métodos de validação apresentam algum viés para algum tipo de partição, normalmente partições com grupos esféricos. Neste trabalho utilizou-se o método *Silhouette*, pelos seguintes motivos:

- é uma medida de fácil leitura, pois está em [-1;1], onde -1 indica um agrupamento mal formado, 1 representa um agrupamento correto e 0 denota partições com clusters sobrepostos;
- consegue validar agrupamentos com grupos coesos (densos) e bem separados, o que é uma definição intuitiva válida para uma boa partição no nosso problema.

4.2.1 O critério silhueta

O critério de silhueta avalia a qualidade do agrupamento verificando a similaridade entre os elementos do mesmo grupo e a distância desses elementos para o grupo mais próximo. É preciso determinar uma função para medir a proximidade entre dois elementos, o que pode ser feito calculando a distância Euclidiana entre os pontos P = $\{p_1, p_2, ..., p_n\}$ e Q = $\{q_1, q_2, ..., q_n\}$, definida na Equação (4.5):

$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + ... + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \tag{4.5}$$

Seja i um indivíduo presente em um *cluster* A, a silhueta $s(i)$ é definida pela Equação (4.6):

$$s(i) = \frac{b(i) - a(i)}{max\{a(i), b(i)\}} \tag{4.6}$$

$a(i)$ é a média de dissimilaridade de i para todos outros indivíduos de A.

Calculando a média de dissimilaridade de i para todos os indivíduos de outro *cluster* C, desde que $C \neq A$, a menor dissimilaridade encontrada para C é definida como $b(i)$ e C é chamado de *cluster* vizinho de i [39].

Rousseeuw, 1987 [39] ressalta que se o *cluster* A contém um único indivíduo, o valor de $s(i)$ deveria ser definido como 0.

O resultado da Equação (4.6) estará no intervalo $[-1, 1]$, onde quanto mais próximo de 1 for esse resultado, o indivíduo i é considerado bem classificado em seu *cluster*. Um valor próximo a -1 indica que o indivíduo i é mais próximo de indivíduos de outro *cluster* e portando mal classificado no seu *cluster atual*.

O silhueta pode ser utilizado para definir o número ideal de k *clusters*, por meio do cálculo da média das silhuetas dos indivíduos de cada *clusters* e posteriormente a silhueta da partição calculando a média da silhueta dos *clusters* [20].

4.3 Tecnologias utilizadas

4.3.1 Plataforma Android

A plataforma Android foi inicialmente desenvolvida em 2003 pela Android Inc., empresa fundada por Andy Rubin, Rich Miner, Nick Sears e Chris White na Califórnia, Estados Unidos, que depois foi comprada pelo Google em 2005. O Android é um Sistema Operacional de código aberto sob a licença Apache, baseado em kernel linux, que nas palavras de Rubin, foi criado para desenvolver "dispositivos móveis mais inteligentes, que estejam mais cientes das preferências e da localização do seu dono" [27]. O Android está presente em mais de 1 bilhão de dispositivos, desde smartphones e tablets, a tvs, relógios com o *Android Wear* e carros com o *Android Auto* [12].

Arquitetura Android

A arquitetura da plataforma está representada na Figura 4.9, ela é dividida em *Linux Kernel*, Bibliotecas, *Android Runtime*, *Application Framework* e Aplicativos.

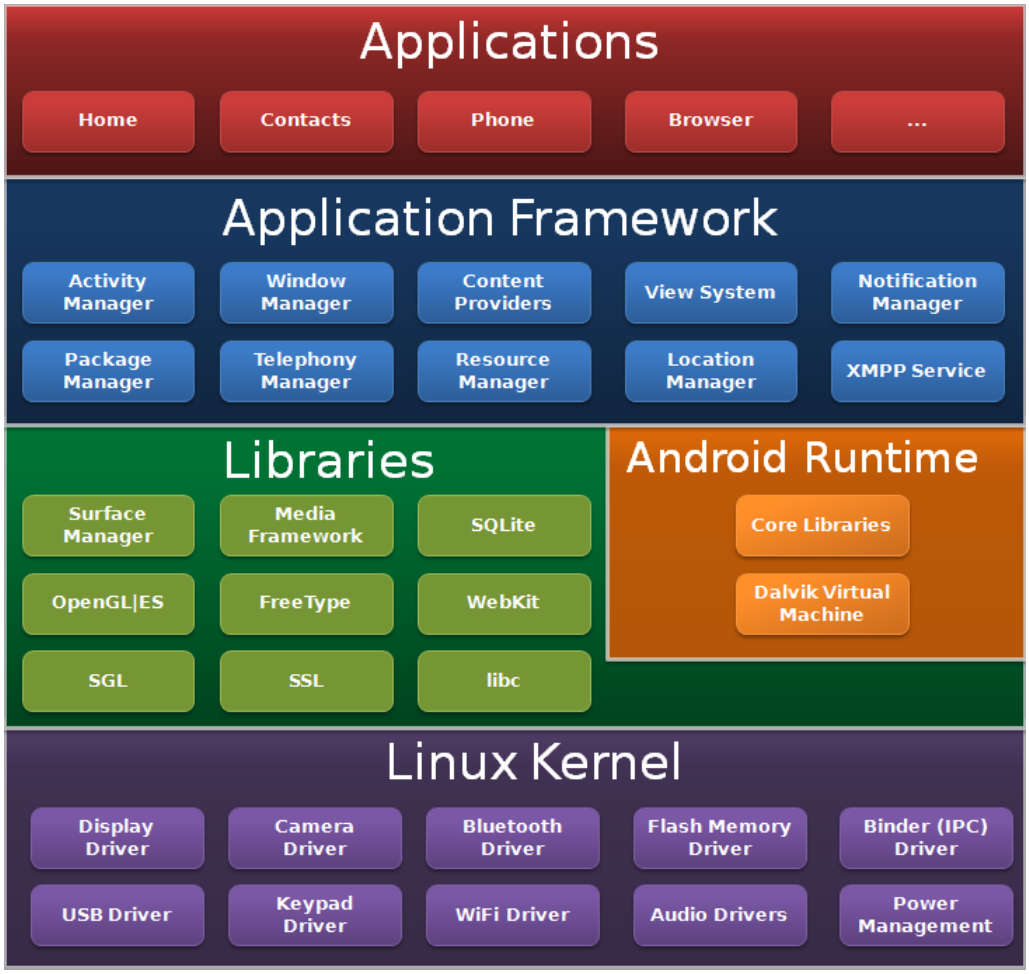


Figura 4.9: Arquitetura da plataforma Android.

Linux Kernel

O *Linux Kernel* encontra-se na camada inferior. O *kernel* é o núcleo de qualquer sistema operacional e é responsável por conter todos os *drivers* essenciais para interagir com o *hardware*. É tido como o coração do sistema. Ele nunca se comunica diretamente com o usuário, mas ofecere uma interface simplificada para que os programas de usuário possam se comunicar com o *hardware*. Algumas das principais funções que ele mantém são: gerenciamento de memória, gerenciamento de energia, gerenciamento de processos, abstração de *hardware*, configurações de segurança, pilha de rede e modelo do *driver*.

Bibliotecas

Nesta camada estão presentes bibliotecas nativas do Android, escritas nas linguagens C ou C++, que contém instruções de como o dispositivo pode manipular diferentes tipos de dados. Abaixo as bibliotecas mais utilizadas e suas funções:

- *Media Framework*: oferece suporte para reprodução e gravação de vários formatos de áudio e vídeo;
- *WebKit*: motor para exibir conteúdo HTML nos navegadores, facilitando o carregamento de páginas;
- *Surface Manager*: disponibiliza acesso ao subsistema de exibição, oferecendo múltiplas camadas gráficas para aplicativos em 2D e 3D;
- *SQLite*: provê um banco de dados relacional para fins de armazenamento de dados, que pode ser usado pelas aplicações diversas ou pelo próprio sistema;
- *SSL*: responsável pela segurança no transporte de pacotes de dados na internet;
- *OpenGL*: oferece suporte para renderização de conteúdos gráficos em 2D ou 3D.

Android Runtime

Compreende dois componentes: *Dalvik Virtual Machine (DVM)* e *Core Java Libraries* (bibliotecas de núcleo Java). Similar a *Java Virtual Machine (JVM)*, mas incapaz de executar arquivos .class, a DVM opera em arquivos .dex, construídos a partir dos arquivos .class como um passo a mais durante a compilação. Por razões de segurança, cada aplicativo no Android é executado em sua própria Máquina Virtual [1], o que proporciona o isolamento e melhor gerenciamento de memória. *Core Java Libraries* oferece a maioria das funcionalidades que estão definidas nas bibliotecas *Java SE*.

Framework de aplicações

Esta camada é de grande importância para os desenvolvedores, pois através dela obtém-se acesso a um grande conjuntos de classes e interfaces que disponibilizam ferramentas para auxiliar a construção das aplicações, a chamada *Java APIs* (do acrônimo *Application Programming Interface) for applications*. A lista a seguir traz alguns serviços providos por essa camada:

- *Activity Manager*: responsável pelo ciclo de vida de cada *activity* que está em execução na aplicação, gerenciando seu estado. Uma *activity* é uma classe que gerencia a interface com o usuário;
- *View*: fornece um conjunto de componentes gráficos que um aplicativo pode utilizar para intergair com o usuário;
- *Resource Manager*: gerencia o acesso a diversos tipos de recursos próprios da aplicação, como imagens, definições de cores, arquivos de layout, *strings*, entre outros;
- *Content Providers*: possibilita a criação de *interfaces* onde uma aplicação pode ter seus dados acessados por outra aplicação;
- *Notification Manager*: gerencia notificações e alertas dos aplicativos;
- *Location Manager*: suporta o acesso aos serviços do aparelho que permitem obter a localização atual, como a leitura de dados do *GPS*;

Aplicativos

A camada superior da arquitetura, onde os aplicativos se encaixam. Aplicações podem ser obtidas por lojas de aplicativos como a *Google Play* ou *Amazon Appstore*. Além disso, os dispositivos trazem uma série de aplicativos que já vêm instalados por padrão, como o cliente de SMS, o Gerenciador de contatos e o Discador.

4.3.2 Eclipse e Plugin ADT

O Eclipse é uma *Integrated Development Environment (IDE)* para desenvolvimento, que apresenta suporte para várias linguagens de programação, seja nativamente ou por meio de *plugins*, sendo a mais popular IDE para desenvolvimento na linguagem Java [15]. É uma ferramenta *open source*, desenvolvida em Java pela *IBM* e que atualmente é mantida pela *Eclipse Foundation*. Foi o ambiente escolhido para desenvolvimento do aplicativo deste trabalho, onde foi utilizada a versão *Luna Service Release 2 (4.4.2)*.

O *ADT (Android Developmente Tools)* é um *plugin* fornecido pelo *Google* que provê ao *Eclipse* um ambiente apropriado para desenvolvimento de aplicações para Android na linguagem Java, adicionando diversas funcionalidades, como pacotes baseados na *Android Framework*

API, depuração dos aplicativos por meio das ferramentas do *SDK Android*, emuladores de aparelhos e assinatura do aplicativo para exportá-lo e distribuí-lo nas lojas de aplicativos.

4.3.3 SQLite

É um banco de dados com suporte nativo no Android, onde os dados são armazenadas em um arquivo em disco, que poderá ser acessado sem a existência de um processo servidor, nem prévia instalação ou configuração. É um banco leve e compacto, a forma mais comum de persistência no Android. Só pode ser acessado pelo aplicativo que o criou, assim como a remoção desse aplicativo resultará na automática remoção do arquivo do banco.

4.3.4 Weka

Waikato Environment for Knowledge Analysis (Weka) é uma ferramenta desenvolvida em Java, que contém diversos algoritmos de aprendizagem de máquina, para a realização de mineração de dados. Provê mecanismos para tarefas de Pré-processamento, Classificação, Regressão, Clusterização, Regras de Associação e Visualização. É um *software open source* desenvolvido na Universidade de Waikato, Nova Zelândia. Utilizado neste trabalho para tarefas de Clusterização e geração do histograma da base de dados.

4.3.5 Biblioteca Colt

Fornece um conjunto de bibliotecas de código aberto escritas em Java, para tarefas de computação científica com alta performance. Alguns dos seus algoritmos são usados para fins de análise de dados, álgebra linear, matriz multidimensionais e estatística. Neste trabalho, foi utilizada como requisito para adicionar ao Weka o algoritmo *Spectral Clustering* baseado na implementação de [41] e [28] , que não estava presente na instalação padrão do Weka.

Capítulo 5

Sistema de recomendação alimentar proposto

O sistema proposto tem como principal finalidade auxiliar o usuário a montar um plano de alimentação, onde os nutrientes dos alimentos estejam dentro de um limite informado pelo usuário, definido por um nutricionista. A escolha dos alimentos para sugestão leva em consideração o seu histórico de consumo, dando preferência a alimentos que o usuário consumiu e indicou como de sua preferência.

É permitido ao usuário a possibilidade de trocar algum alimento do plano sugerido, onde ao acionar essa opção, o sistema disponibiliza uma lista com novos alimentos sugeridos que tenham suas informações nutricionais similares ao alimento a ser substituído e assim não comprometam o limite da dieta, ficando a escolha na lista a critério do usuário.

Foram utilizados métodos de clusterização na base de dados, para formar grupos de alimentos com valores nutricionais similares. Assim, facilitando o processo de troca de alimentos no plano sugerido, onde a lista sugerida é formada com outros alimentos no mesmo grupo do alimento a ser substituído.

A Figura 5.1 representa o diagrama de casos de uso com as principais funcionalidades do sistema, que serão detalhadas nas seções a seguir. A tela inicial do aplicativo é apresentada na Figura 5.2.

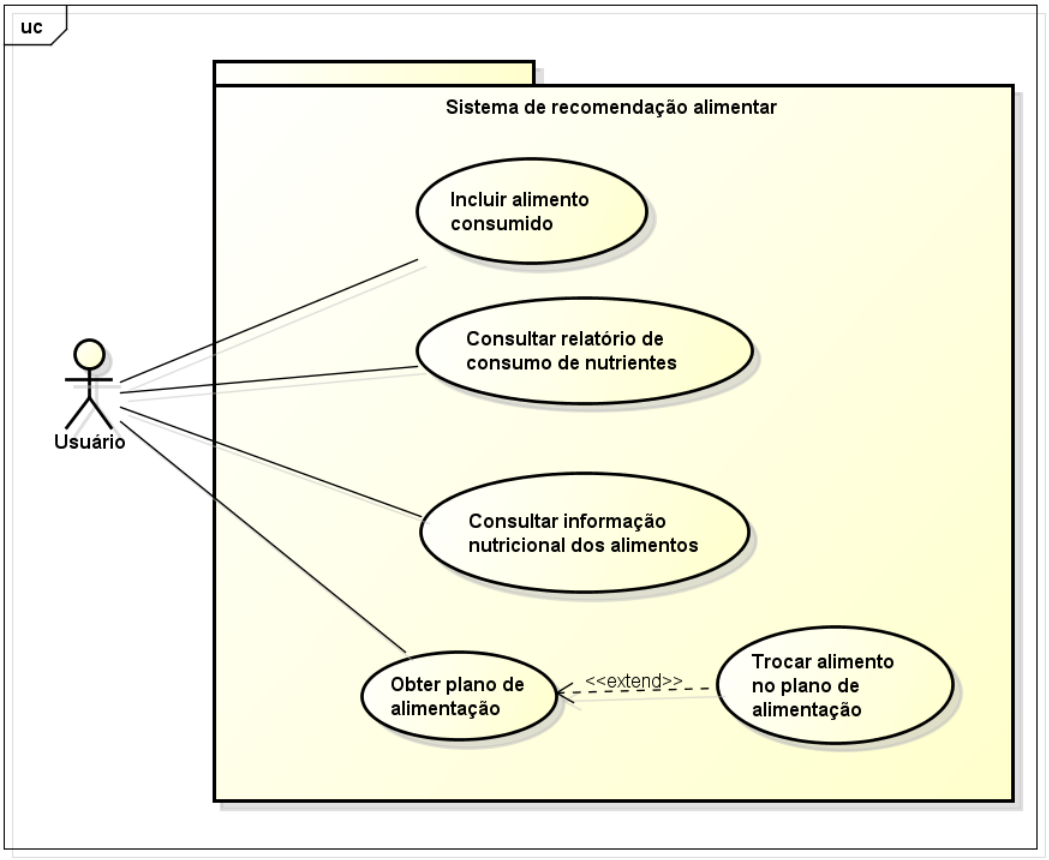


Figura 5.1: Diagrama de casos de uso

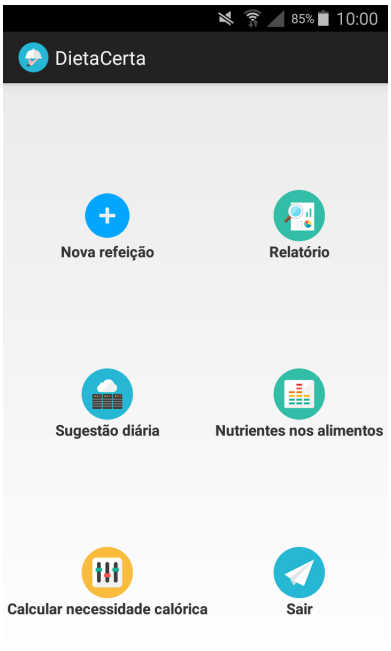
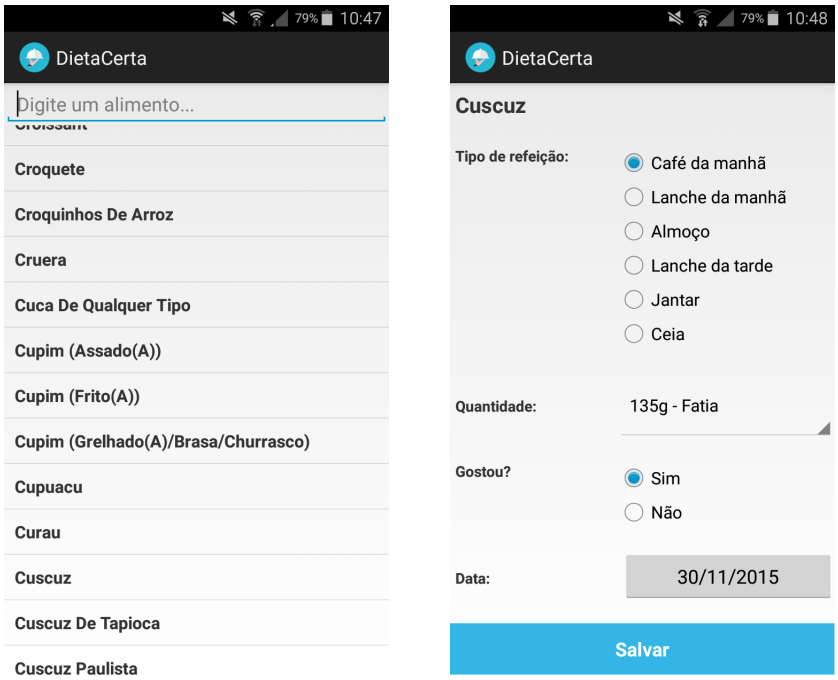


Figura 5.2: Tela inicial

5.1 Inserir alimento consumido

Esta funcionalidade é dividida em 2 passos. No primeiro, demonstrado na Figura 5.3(a), o usuário pesquisa qual alimento que ele consumiu. No passo seguinte, Figura 5.3(b), ele informa em qual refeição o alimento foi consumido, a data e se ele gostou. O sistema define a quantidade em gramas de acordo com a porção padrão para o alimento que foi escolhido, mas fica a critério do usuário alterar essa quantidade para outras definições de porções ou um valor desejado.

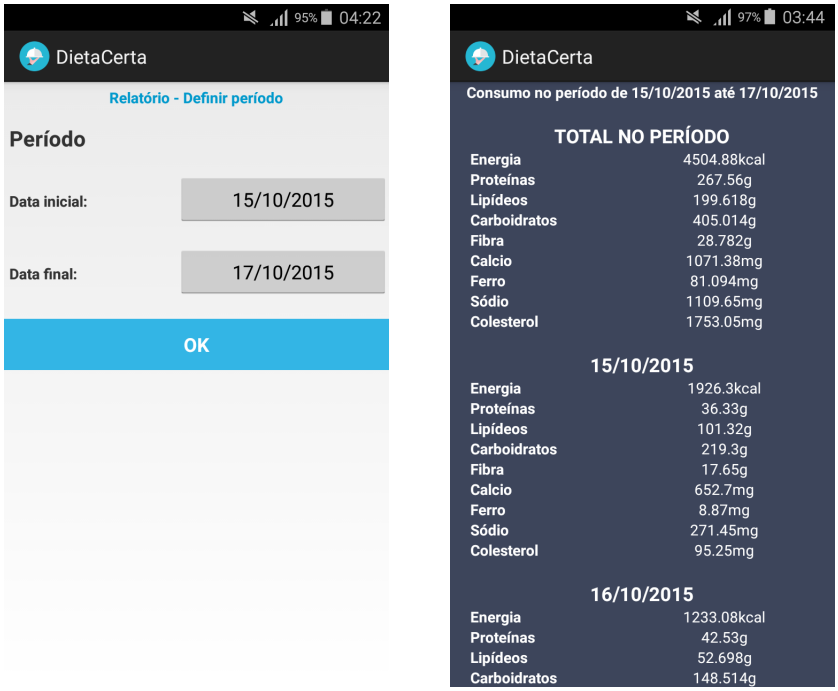


(a) Escolher alimento (b) Confirmar dados da refeição

Figura 5.3: Nova refeição

5.2 Relatório de consumo

Com o relatório, o usuário pode manter-se informado se suas metas de consumo de nutrientes para um dado período estão sendo cumpridas. Esta funcionalidade é representada nas Figuras 5.4(a) e 5.4(b).

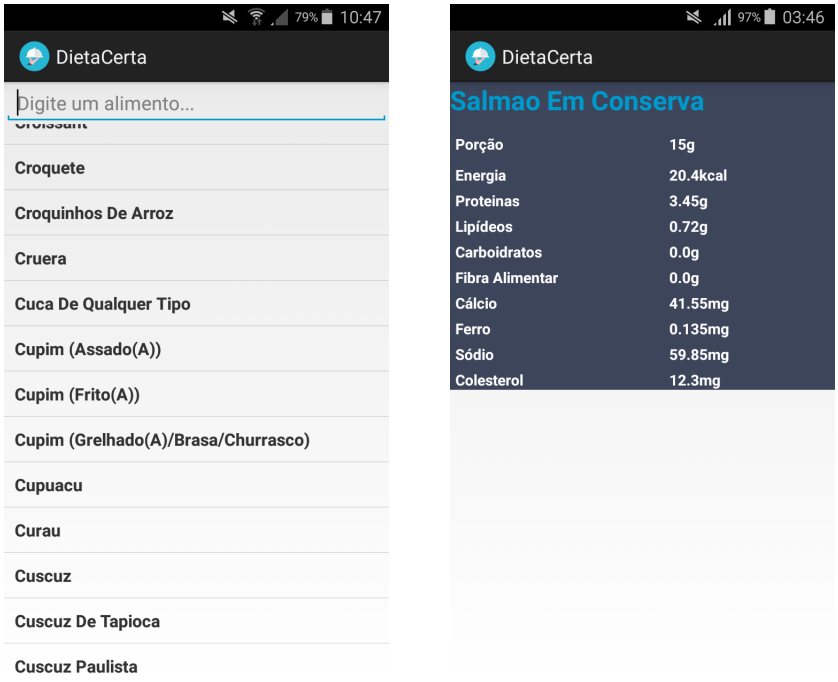


(a) Período do relatório (b) Relatório

Figura 5.4: Relatório de nutrientes consumidos

5.3 Nutrientes nos alimentos

Aqui o usuário poderá obter a informação nutricional de todos os alimentos que estão presentes na base de dados. As Figuras 5.5(a) e 5.5(b) representam essa funcionalidade.



(a) Lista de alimentos (b) Nutrientes do alimento

Figura 5.5: Nutrientes do alimento

5.4 Cálculo da necessidade calórica diária

Uma maneira de determinar a ingestão calórica de cada perfil de usuário é através da fórmula de *Recomendação Estimada de Energia* [31], que utiliza no cálculo as variáveis sexo, idade, altura e nível de atividade física.

A Equação (5.1) apresenta a fórmula para o sexo masculino, enquanto a fórmula para o sexo feminino é representada na Equação (5.2):

$$Sexo_{masculino} : REE = 662 - 9,53 * Idade + AF * (15,91 * Peso + 539,6 * Altura) \quad (5.1)$$

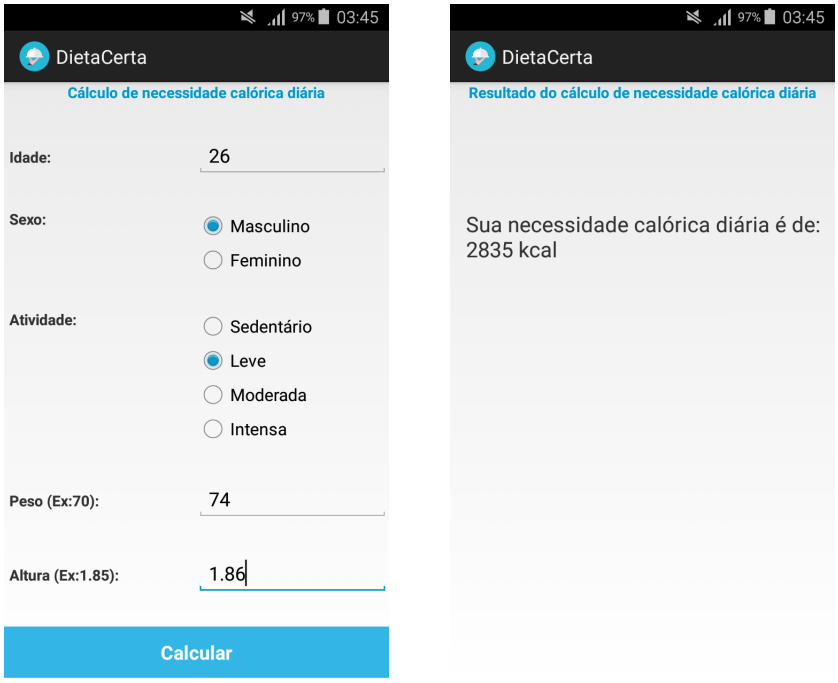
$$Sexo_{feminino} : REE = 354 - 6,91 * Idade + AF * (9,36 * Peso + 726 * Altura) \quad (5.2)$$

onde AF representa o nível de atividade física, conforme a Tabela 5.1.

Nível de atividade física	Coeficiente de atividade física (FA)	
	Masculino	Feminino
Sedentário	1	1
Leve	1.11	1.12
Moderada	1.25	1.27
Intensa	1.48	1.45

Tabela 5.1: Coeficiente de atividade física

As Figuras 5.6(a) e 5.6(b) representam esta funcionalidade.

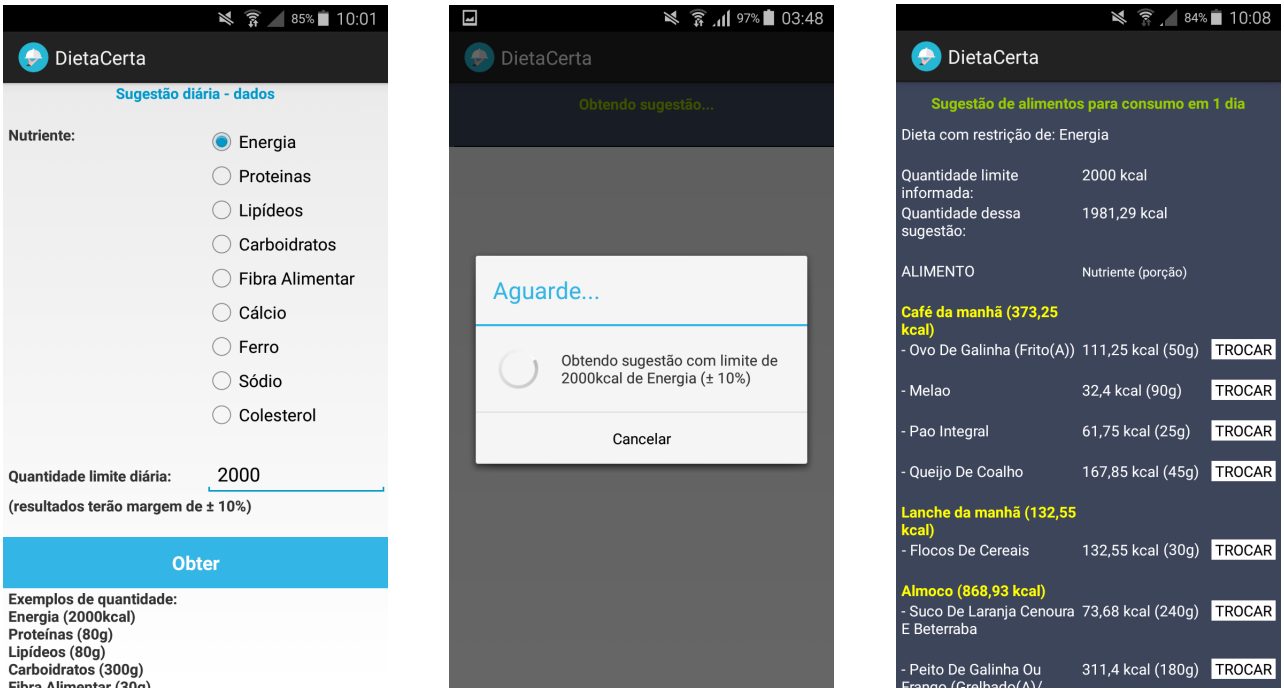


(a) Dados do usuário (b) Resultado do cálculo

Figura 5.6: Cálculo da necessidade calórica diária

5.5 Sugestão de plano de alimentação

Na tela apresentada na Figura 5.7(a), o usuário informa qual nutriente ele deseja estabelecer um valor limite. E então, após o sistema encontrar uma combinação de alimentos que seja adequada ao valor informado, uma sugestão de plano de alimentação é apresentada, como mostrado na Figura 5.7(c).



(a) Nutriente e limite (b) Processamento (c) Sugestão de alimentos

Figura 5.7: Sugestão de alimentação diária

Optou-se por somente considerar os resultados em que a soma total do nutriente no plano sugerido estivesse entre 90% e 110% do valor limite informado, para que o algoritmo não retornasse uma combinação que estivesse muito abaixo do limite. Isso poderia trazer alimentos com grande escassez do nutriente informado e acabar não sendo do agrado do usuário. Sendo o valor limite um referencial para o plano de alimentação sugerido.

Para demonstrar como a sugestão é feita, o Algoritmo 1 apresenta o pseudocódigo com um escopo mais reduzido, buscando apenas alimentos para o Almoço com um limite de Energia (calorias), mas a busca por outras refeições ocorre de maneira similar. A quantidade de calorias limite informada pelo usuário é distribuída ao longo das refeições do dia, com a proporção sugerida por [2] e apresentada na Tabela 5.2.

Refeição	**Proporção de calorias**
Café da manhã	15%
Lanche da manhã	5%
Almoço	35%
Lanche da tarde	5%
Jantar	30%
Ceia	10%

Tabela 5.2: Distribuição de calorias sugeridas para as refeições ao longo do dia. (adaptado) [2]

Algoritmo 1: Sugestão de alimentos para o Almoço com limite de Energia

Entrada: LimiteEnergiaAlmoco

Saída: Lista de alimentos para almoço, dentro do limite de energia

inicio

alimentosAlmoco = Todos os alimentos para almoco;

alimentosAlmocoConsumidos = alimentos consumidos no almoço que o usuário gostou;

para $i \in$ *alimentosAlmocoConsumidos* **faca**

adiciona i a alimentosAlmoco;

fim

quantidadeAlimentos = 0;

energiaAlmoco = 0;

sugestaoAlimentosAlmoco = [ ];

enquanto *energiaAlmoco* < *limiteEnergiaAlmoco * 0.9* **faça**

se *quantidadeAlimentos* > *7* **então**

quantidadeAlimentos = 0;

energiaAlmoco = 0;

sugestaoAlimentosAlmoco = [ ];

fim

alimentoSorteado = item aleatório em alimentosAlmoco;

porcaoPadrao = busca porção padrão (alimentoSorteado);

energiaPorcaoPadrao = (porcaoPadrao * alimentoSorteado.Energia) / 100

Os alimentos na base têm seus valores nutricionais relativos a uma porção de 100g ### ;

se (*energiaPorcaoPadrao + energiaAlmoco*) > (*limiteEnergiaAlmoco * 1.1*)

então

continue;

fim

adicionar alimentoSorteado a sugestaoAlimentosAlmoco;

quantidadeAlimentos += 1;

fim

fim

5.6 Troca de alimento no plano de alimentação

Com o objetivo de criar conjuntos de alimentos, agrupados pela similaridade, a fim de disponibilizar para o usuário do aplicativo a opção de trocar um alimento da sugestão por outro que fosse nutricionalmente parecido, nesta etapa foi feito a clusterização da base de dados, executando cada algoritmo na base de dados com o mesmo estado inicial, para salvar seu resultado e poder determinar posteriormente qual algoritmo obteve o melhor particionamento, por meio do cálculo do Silhouette. Os algoritmos usados foram: K-means, X-means, Expectation Maximization e Spectral Clustering. Exceto o Spectral, os outros algoritmos permitem a definição do número de clusters que devem ser encontrados na solução. A cada um deles, foi obtido o resultado do particionamento com k clusters, sendo k valores do intervalo de 5 à 100.

O algoritmo Spectral Clustering não permite a definição do número de grupos que serão formados, então foi feito um *grid search* com seus parâmetros α^* (controla o corte das arestas) e σ (controla a afinidade entre dois pontos de dados), com a finalidade de se obter resultados com a quantidade de clusters dentro do intervalo entre 5 e 100. Os valores definidos para esses parâmetros e a quantidade de clusters obtidos são mostrados na Tabela 5.3. Na Tabela 5.4 é mostrado o cálculo do Silhouette para estes particionamentos gerados que tiveram mais de 1 cluster.

		alphaStar						
		0,001	**0,01**	**0,1**	**0,2**	**0,5**	**0,75**	**0,9**
sigma	**0,01**	431	514	582	614	682	724	750
	0,05	35	53	94	125	222	350	484
	0,1	11	17	30	33	53	121	285
	1	1	1	1	1	1	1	1
	10	1	1	1	1	1	1	1
	100	1	1	1	1	1	1	1
	1000	1	1	1	1	1	1	1

Tabela 5.3: Experimento com Spectral Clustering - quantidade de clusters gerados

		alphaStar						
		0,001	**0,01**	**0,1**	**0,2**	**0,5**	**0,75**	**0,9**
sigma	**0,01**	0,521	0,72	0,848	0,878	0,94	0,969	0,986
	0,05	0,131	0,304	0,327	0,363	0,493	0,602	0,749
	0,1	0,417	0,309	0,496	0,456	0,442	0,423	0,528

Tabela 5.4: Experimento com Spectral Clustering - cálculo do Silhouette

A Figura 5.8 apresenta o cálculo do Silhouette aplicado aos 4 algoritmos. Os pontos de descontinuidade no gráfico são devido a não se obter resultados com a quantidade de clusters definida no eixo X.

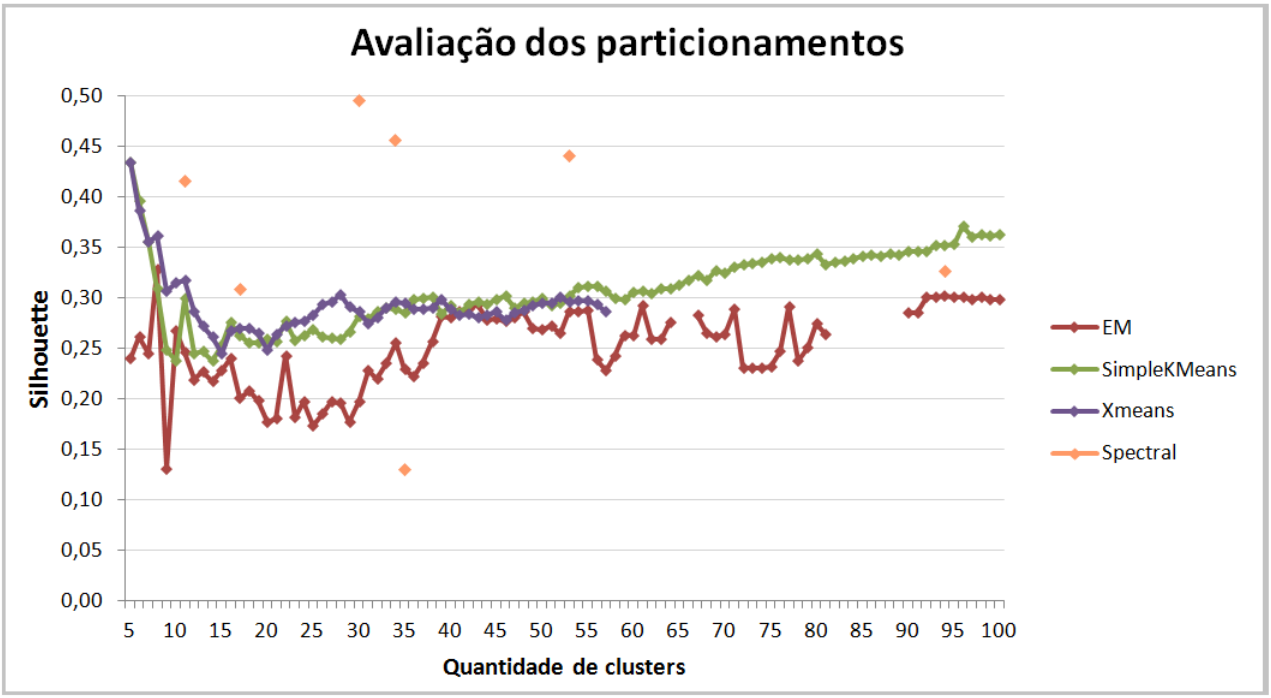


Figura 5.8: Cálculo do Silhouette para os particionamentos encontrados utilizando K-means, X-means, EM e Spectral Clustering

Para a escolha do melhor particionamento, além de considerar o melhor valor de Silhouette, optou-se por também considerar os particionamentos onde os clusters apresentassem em média a mesma quantidade de alimentos. Neste cenário, o algoritmo Spectral Clustering apresentou desvantagem, pois as partições geradas tinham clusters com 80% dos alimentos. Os demais clusters continham menos de 5 alimentos, sendo muitas vezes o mesmo alimento

nos seus diferentes modos de preparo ou um pequeno grupo de alimentos bastante restrito.

O Silhouette dá nota maior para partições com grupos mais coesos, mas clusters com apenas 1 alimento em suas diferentes formas de preparo não dariam opções para o usuário efetuar a troca. Diante disso, o algoritmo K-means mostrou-se mais eficaz, por apresentar clusters que, além de achar grupos de alimentos restritos, incrementa a quantidade de instâncias no grupo, dando mais opções e diversidade para o usuário efetuar a troca de alimentos.

Com o comprimisso entre a medida Silhouette e a diversidade de alimentos em um cluster, foi escolhida a partição com 96 clusters gerada pelo K-means.

Após os experimentos descritos acima e escolha do particionamento mais adequado ao nosso problema, a tabela de alimentos na base de dados recebeu um atributo *cluster* que indica a qual grupo cada alimento pertence.

Com o plano de alimentação apresentado ao usuário, ele tem a opção de trocar alguns alimentos sugeridos, caso não seja do seu agrado naquele momento. Ao clicar no botão *TROCAR* de algum alimento, as seguintes variáveis são iniciadas:

- *cluster*: recebe a identificação do cluster do alimento a ser substituído;
- *tipoRefeicao*: indica em que refeição acontecerá a troca;
- *margemInferior*: indica qual o valor mínimo que o novo alimento deve conter para o nutriente, para que o plano de alimentação ainda esteja a pelo menos 90% da meta diária;
- *margemSuperior*: indica qual o valor máximo que o novo alimento deve conter para o nutriente, para que o plano de alimentação não exceda em mais de 10% da meta diária;

Então é feito uma busca na base de dados por alimentos para a mesma refeição e que estejam no mesmo *cluster*, que não ultrapassem as margens inferior e superior que foram salvas. O resultado dessa busca é mostrado ao usuário em forma de lista, onde ele poderá escolher algum alimento a seu critério para efetuar a troca Figura 5.9.

Caso a troca aconteça, são recalculados os novos valores para a quantidade total do nutriente no plano de alimentação e a quantidade na refeição onde houve a troca.

Figura 5.9: Troca de alimentos da sugestão

Capítulo 6

Considerações finais

Neste trabalho foi desenvolvido um sistema de recomendação alimentar para a plataforma Android, com o objetivo de auxiliar o usuário a montar um plano de alimentação adequado a um valor limite estabelecido para algum nutriente. Utilizando técnicas de agrupamento, foi possível encontrar alimentos similares na base de dados e assim fornecer um método inteligente de sugestão de novos alimentos, quando o usuário optar por trocar algum alimento no plano sugerido.

6.1 Conclusão do trabalho

O principal motivo que leva as pessoas a abandonarem as dietas tradicionais é a facilidade de se entediar de um mesmo alimento que está se comendo várias vezes. Neste sentido, o aplicativo visa aperfeiçoar a dieta ao trazer uma grande quantidade de combinações possíveis, como também levar o usuário a conhecer novos alimentos, levando a diminuição do índice de abandono da dieta.

Por meio de relatos de usuários, foi possível constatar que a eficácia do aplicativo está em trocar os alimentos da sugestão sem comprometer a dieta. A disponibilização do sistema por meio de aplicativo móvel, tornando-o acessível a qualquer momento, mesmo quando o usuário não têm uma conexão com a internet, também foi uma das vantagens mais relatadas.

Por outro lado, a sugestão de alimentos com base no gosto do usuário não pôde ser avaliada, pelo curto espaço de tempo entre a disponiblização da versão final do aplicativo e o período

de avaliação dos usuários.

6.2 Trabalhos futuros

Algumas propostas de melhorias no sistema são:

- Possibilitar que o usuário mantenha um cadastro de metas de consumo dos nutrientes e que o relatório de consumo também faça uma comparação entre as metas do usuário e o status atual de consumo;
- Informar ao usuário por meio de alertas assim que ele exceder seu limite diário máximo de algum nutriente;
- Traduzir tabelas de alimentos encontradas em outros idiomas, para posterior inclusão na base de dados;
- Permitir que o usuário possa remover alimentos da base de dados;
- Disponibilizar o sistema em outras plataformas como iOS e Windows Phone;
- Utilizar técnicas de otimização multiobjetivo para sugerir planos de alimentação que tenham mais de 1 nutriente com valores limites estabelecidos.

Referências Bibliográficas

[1] A. Dubey e A. Misra. *Android Security: Attacks and Defenses*. Auerbach Publications, Boca Raton - Flórida - Estados Unidos, 2013.

[2] C. Antonaccio. *Boa forma em 8 semanas*. Editora Abril, São Paulo, 2006.

[3] T. Calinski e J. Harabasz. A dendrite method for cluster analysis. *Communications in Statistics*, 3:1–27, 1974.

[4] L. M. Carlantonio. *Novas metodologias para clusterização de dados*. Tese, COPPE, Universidade Federal do Rio de Janeiro, Rio de Janeiro - RJ, Setembro 2001.

[5] L. C. Coradine, R. V. V. Lopes e A. F. Maciel. Mineração de dados: Uma introdução. *Journal of the Brazilian Neural Network Society*, 9:168–184, 2011.

[6] D. L. Davies e D. W. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2:224–227, 1979.

[7] Universidade Estadual de Campinas (UNICAMP). TACO (Tabela Brasileira de Composição de Alimentos). Acessado em 05/12/2015. URL: `www.unicamp.br/nepa/taco/`.

[8] Instituto Brasileiro de Geografia e Estatística (IBGE). Tabela de composição nutricional dos alimentos consumidos no Brasil. Acessado em 05/12/2015. URL: `www.ibge.gov.br/home/estatistica/populacao/condicaodevida/pof/2008_2009_composicao_nutricional/default_zip.shtm`.

[9] A. P. Dempster, N. M. Laird e D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39:1–38, 1977.

[10] J. Dunn. Well separated clusters and optimal fuzzy partitions. *Journal of Cybernetics*, 4:95–104, 1974.

[11] Dieta e saúde. Acessado em 05/12/2015. URL: `www.dietaesaude.com.br`.

[12] Android. Acessado em 05/12/2015. URL: `www.android.com`.

[13] Tecnonutri Gestão em alimentação. Acessado em 05/12/2015. URL: `www.tecnonutri.com.br`.

[14] D. T. Junior et al. Sistema de raciocínio baseado em casos para recomendação de programa alimentar. *RESI – Revista Eletrônica de Sistemas de Informação*, 3, 2006.

[15] J. Kabanov et al. Developer productivity report 2013. Relatório técnico, RebelLabs, 2013.

[16] M. E. Nita et al. *Avaliação de tecnologias em saúde: Evidência Clínica, Análise Econômica e Análise de Decisão.* Artmed, Porto Alegre, 2010.

[17] P. C. Emiliano et al. Fundamentos e comparação de critérios de informação: Akaike and bayesian. *Rev. Bras. Biom.*, 27:394–411, 2009.

[18] T. John et al. A mobile food recommendation system based on the traffic light diet. *University of Arizona*, 2014.

[19] K. Faceli. *Um framework para análise de agrupamento baseado na combinação multi-objetivo de algoritmos de agrupamento.* Tese, USP, São Carlos, Janeiro 2007.

[20] A. Fontana e M. C. Naldi. Estudo e comparação de métodos para estimação de números de grupos em problemas de agrupamento de dados. Relatório técnico 340, Instituto de Ciências Matemáticas e de Computação, São Carlos, Março 2009.

[21] G. Franco. *Tabela de Composição Química dos Alimentos.* Editora Atheneu, Rio de Janeiro, 2005.

[22] V. O. Gil, L. Emmendorfer e F. Ferrari. Análise de agrupamentos para classificação morfométrica de galáxias. *XVII Encontro de Modelagem Computacional*, 2014.

[23] R. Goldshmidt, E. Bezerra e E. Passos. *Data Mining: conceitos, técnicas, algoritmos, orientações e aplicações.* Elsevier, Rio de Janeiro, 2015.

[24] A. Halpern. *Dieta dos Pontos Para Viver Mais.* Editora Abril, São Paulo, 2014.

[25] J. Han, M. Kamber e J. Pei. *Data Mining: Concepts and Techniques.* Morgan Kaufman, Waltham - MA, 2012.

[26] Z. Huang. Clustering large data sets with mixed numeric and categorical values. *Proceedings of the 1st Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 21–34, 1997.

[27] D. Cummings e I. Krajci. *Android on X86: An Introduction to Optimizing for Intel® Architecture*. Apress, Califórnia - Estados Unidos, 2013.

[28] R. Kannan, S. Vempala e A. Vetta. On clusterings - good, bad and spectral. Relatório técnico, CS Dept., Yale University, 2015.

[29] R. Liden. *Algoritmos Genéticos: Uma importante ferramenta da Inteligência Computacional*. Brasport, Rio de Janeiro - RJ, 2008.

[30] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *5-th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[31] IOM (INSTITUTE OF MEDICINE). *Dietary reference intakes for energy, carbohydrate, fiber, fat, fatty acids, cholesterol, protein, and amino acids*. National Academies Press, Washington, 2005.

[32] M. C. V. Nascimento. *Metaheurísticas para o problema de agrupamento de dados em grafo*. Tese, ICMC, USP, São Carlos, Fevereiro 2010.

[33] A. Y. Ng, M. I. Jordan e Y. Weiss. On speectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14*, pages 849–856, 2001.

[34] World Health Organization. Obesity and overweight. Acessado em 05/12/2015. URL: `www.who.int/mediacentre/factsheets/fs311/en`.

[35] M. Pacheco. *Tabela de Equivalentes, Medidas Caseiras e Composição Química dos Alimentos*. Editora Rubio, Rio de Janeiro, 2011.

[36] D. Pelleg e A. Moore. X-means: extending k-means with efficient estimation of the number of clusters. *Proceedings of 17th International Conference on Machine Learning*, pages 727–734, 2000.

[37] S. T. Philippi. *Tabela de composição de alimentos*. Manole, Barueri - SP, 2012.

[38] J. A. Quilici-Gonzalez e F. A. Zampirolli. *Sistemas Inteligentes e Mineração de Dados*. Triunfal Gráfica e Editora, Santo André, 2014.

[39] P. J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987.

[40] M. Runo. Foodroid: A food recommendation app for university canteens. *Swiss Federal Institute of Technology Zurich*, 2011.

[41] J. Shi e J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:888–905, 2000.

[42] Statista. Statistics and facts about App Stores. Acessado em 05/12/2015. URL: `www.statista.com/topics/1729/app-stores`.

[43] E. P. Veiga e M. J. F. Vivanco. A medida l como critério de comparação de modelos: uma revisão da literatura. *Rev. Bras. Biom.*, 30:343–352, 2012.

[44] Vigitel. Vigilância de fatores de risco e proteção para doenças crônicas por inquérito telefônico. Relatório técnico, Ministério da Saúde, 2015.

[45] C. Junjie e W. Huiqing. A semi supervised spectral clustering algorithm based on rough sets. In *CASON '10 Proceedings of the 2010 International Conference on Computational Aspects of Social Networks*, pages 344–347, 2010.