



Projeto Final de Programação – INF2102

Pós-Graduação em Informática, PUC-Rio

Hamsa Survey

Ferramenta de limpeza e preparação semi-supervisionada de dados para segmentação de pesquisas exploratórias qualitativas

31/Agosto/2020

Nome Bruno Ferraz de Melo
Email bferraz@inf.puc-rio.br | brunoferraz.pro@gmail.com
Matrícula 1912733
Orientador Bruno Feijó



Table of Contents

1	INTRODUÇÃO	4
1.1	FINALIDADE	4
1.2	CONVENÇÕES DO DOCUMENTO	4
1.2.1	ACRÔNIMOS	4
1.2.2	GLOSSÁRIO	4
1.3	ESCOPO DO PROJETO	4
2	ESPECIFICAÇÃO DE REQUISITOS DO SISTEMA	5
2.1	DESCRIÇÃO GERAL	5
2.1.1	PERSPECTIVA DO PRODUTO	5
2.1.2	CARACTERÍSTICA DO PRODUTO E DO USUÁRIO	6
2.1.2.1	Sobre o Produto	6
2.1.2.2	Sobre o usuário	6
2.1.3	AMBIENTE OPERACIONAL	6
2.1.4	RESTRIÇÕES DE PROJETO E IMPLEMENTAÇÃO	7
2.1.4.1	Linguagem	7
2.1.4.1.1	Python 3	7
2.1.4.2	Bibliotecas	7
2.1.4.2.1	Pandas	7
2.1.4.2.2	Numpy	7
2.1.4.2.3	Scikitlearn	7
2.1.4.3	GUI	7
2.1.4.3.1	PyQT	7
2.1.4.4	IDE	7
2.1.4.4.1	MatLab	7
2.1.4.4.2	QtDesigner	7
2.1.4.5	Ferramentas	7
2.1.4.5.1	Som – ToolBox 2.1	7
2.1.4.5.2	Sphinx	7
2.1.5	DOCUMENTAÇÃO PARA USUÁRIOS	8
2.1.6	HIPÓTESES E DEPENDÊNCIAS	8
2.2	CARACTERÍSTICA DO SISTEMA	8
2.2.1	DESCRIÇÃO	8
2.2.2	CASOS DE USO	9
2.2.3	REQUISITOS FUNCIONAIS	9
2.2.3.1	[RF01]- O sistema deve processar a planilha resultante do google form	9
2.2.3.2	[RF02]- O sistema deve identificar quantas e quais tipos de perguntas e respostas	10
2.2.3.3	[RF03]- O sistema deve verificar se as respostas se adequam ao tipo de resposta esperado	10
2.2.3.4	[RF04]- O sistema deve localizar as instâncias problemáticas (respostas ausentes)	10
2.2.3.5	[RF05]- O sistema deve exibir relatório	10
2.2.3.6	[RF06]- O sistema deve oferecer a opção para ajuste manual ou a retirada da instância	10
2.2.3.7	[RF07]- O sistema deve oferecer a possibilidade de exclusão de colunas (perguntas)	10
2.2.3.8	[RF08]- O sistema deve mostrar a distribuição dos dados no espaço (scatter plot) e oferecer a possibilidade da criação de categorias quando necessário.	10
2.2.3.9	[RF09]- O sistema deve oferecer ao usuário a escolha do tipo de codificação disponível para cada questão	10
2.2.3.10	[RF10]- O sistema deve exportar os dados compatíveis para funcionar como entrada na rede neural	10
2.2.4	REQUISITOS DE DADOS	10
2.2.4.1	Logical data model	10
2.2.4.1.1	Entrada	10
2.2.4.1.2	Processo	11
2.2.4.1.3	Saída	11
2.2.4.2	Data Dictionary	11
2.2.4.3	Reports	11
2.2.4.4	Data Aquisition, Integrity, Retention and Disposal	12

2.3	REQUISITOS DE INTERFACES EXTERNAS	12
2.3.1	INTERFACES DO USUÁRIO.....	12
2.3.1.1	Questionário	12
2.3.1.2	Questão	12
2.3.1.3	Instância.....	12
2.3.1.4	Inspeção.....	13
2.3.2	INTERFACE DE SOFTWARE.....	13
2.4	OUTROS REQUISITOS NÃO FUNCIONAIS	14
2.4.1	USABILIDADE.....	14
2.4.2	CONSISTÊNCIA	14
2.4.3	EXTENSIBILIDADE.....	14
2.4.4	RECUPERABILIDADE.....	14
3	<u>ARQUITETURA DE SOFTWARE</u>	<u>15</u>
3.1	INTRODUÇÃO.....	15
3.2	DETALHAMENTO TÉCNICO DOS REQUISITOS NÃO-FUNCIONAIS	15
3.3	USABILIDADE.....	15
3.3.1	[RNF1] UTILIZAR HINTS.....	15
3.3.2	[RNF2] NAVEGAÇÃO PARA EDIÇÃO	15
3.3.3	[RNF3] NAVEGAÇÃO PARA INSPEÇÃO.....	15
3.4	CONSISTÊNCIA	15
3.4.1	[RNF4] INTERFACE	15
3.5	EXTENSIBILIDADE	15
3.5.1	[RNF5] MODULARIDADE DO TIPO DE QUESTÃO	15
3.5.2	[RNF6] MODULARIDADE DO TIPO DE CODIFICAÇÃO	15
3.5.3	[RNF7] MODULARIDADE DO TIPO DE NORMALIZAÇÃO.....	15
3.6	RECUPERABILIDADE	15
3.6.1	[RNF8] DADOS PERSISTENTES	15
3.7	PERFORMANCE	15
3.8	COMPONENTES DE SOLUÇÃO	15
3.8.1	FAÇADE	16
3.8.2	FACTORY	16
3.8.3	STATE.....	16
3.8.4	DECORATOR.....	17
3.9	VISÃO DE IMPLANTAÇÃO.....	17
3.10	CONTROLE DE VERSÃO.....	17
3.11	BRANCHES	18
3.12	COMMITTS	18
4	<u>TESTES.....</u>	<u>19</u>
4.1	VERIFICAÇÃO E VALIDAÇÃO	19
4.2	SITUAÇÃO ATUAL	19
4.3	REQUISITOS FUNCIONAIS	20
4.4	REQUISITOS NÃO FUNCIONAIS	20
5	<u>REFERENCES</u>	<u>20</u>

1 Introdução

1.1 Finalidade

A ferramenta Hamsa é pensada para ajudar pesquisadores de áreas não tecnológicas no processo de limpeza dos dados obtidos por formulários online e prepará-los para serem processados em redes neurais para segmentação. Com esta ferramenta, o pesquisador usa uma planilha de respostas, como entrada, obtida através do *google form* e o sistema prepara de forma supervisionada os dados para serem processados por técnicas de redes neurais para segmentação, tais como mapa de kohonen e k-means. Na sua versão inicial, esta ferramenta deve funcionar entre o *google forms* e o *Matlab*.

Esta ferramenta está sendo desenvolvida como um componente necessário para as atividades de pesquisa do doutorado do presente aluno, visto que ele deve tratar de um grande volume de dados obtidos através de surveys. Esta ferramenta deve agilizar e automatizar a criação de testes e extração de informação para análises posteriores.

No projeto final de programação (INF2102), a concepção geral do sistema é considerada, porém apenas um subconjunto dos módulos e parte das funcionalidades são implementadas. O intuito é mostrar o domínio dos métodos e técnicas de programação. O desenvolvimento completo da ferramenta deve continuar como parte das atividades da pesquisa de doutorado.

Alguns dos principais requisitos apresentados neste documento referem-se às diferentes demandas e metodologias dos domínios envolvidos.

1.2 Convenções do documento

1.2.1 Acrônimos

TBD – To be defined

1.2.2 Glossário

Hamsa significa cisne em sânscrito. O animal foi escolhido para batizar o sistema devido a lenda que o aponta como capaz de separar a água do leite. Comumente é utilizado como símbolo de discernimento e veículo do conhecimento. Cabendo assim, no contexto de uma ferramenta para limpeza de dados e mineração de dados.

dataFrame Principal estrutura de dados da biblioteca Pandas

Instância Entidade relativa a cada entrada de respostas em um questionário. Conjunto de todas as respostas que um entrevistado forneceu

Label Nome curto utilizado para identificar cada questão no questionário e na rede neural

Usuário Quem utiliza o Hamsa

Entrevistado Quem respondeu ao questionário

Pandas Biblioteca do python para manipulação de dados

1.3 Escopo do Projeto

O desenvolvimento do sistema Hamsa adotará o modelo de processo evolucionário incorporando paradigma de prototipação. Essa escolha se deve aos requisitos herméticos do domínio que naturalmente se evidenciam mediante situações concretas que ajudam não só na elicitación de novos requisitos, mas também na validação dos anteriormente definidos.

Em alto nível, a primeira versão da ferramenta deverá apresentar as seguintes funcionalidades:

1. Processar a planilha resultante do google form
2. Identificar quantas e quais tipos de perguntas e respostas
3. Verificar se as respostas se adequam ao tipo de resposta esperado
4. Localizar as instâncias problemáticas e oferecer o ajuste manual ou a retirada da instância
5. Possibilidade de exclusão de colunas (perguntas)
6. Mostrar a distribuição dos dados no espaço (scatter plot) e oferecer a possibilidade da criação de categorias quando necessário.
7. Oferecer a possibilidade de escolha para o tipo de codificação
8. Exportação dos dados como entrada para rede neural

2 Especificação de requisitos do Sistema

2.1 Descrição geral

Essa seção apresentará uma visão geral em alto nível do produto, do ambiente em que ele será usado e possíveis usuários. Relatará, também dependências, possibilidades e restrições conhecidas.

2.1.1 Perspectiva do produto

Nos últimos anos, ferramentas que automatizam o processo de criação e aplicação de formulários online popularizaram-se no meio acadêmico de áreas não tecnológicas agilizando certas etapas da pesquisa e gerando gargalos em outras. Dessa forma, o aumento massivo de dados, muitas vezes, é tão grande/complexa que a análise manual se torna impraticável e gera a necessidade de técnicas de redes neurais para também automatizar parte da análise ou extrair *features* dos dados para visualização. Nesse contexto, surge um hiato entre a realidade dados disponíveis e como os dados precisam estar para funcionarem como entradas em tais técnicas. Principalmente quando os especialistas dos domínios em questão não dominam o ferramental disponível.

Inicialmente, o sistema funcionará como uma interface de comunicação entre os resultados do google form e a entrada para o pacote SOM do matlab.

2.1.2 Característica do produto e do usuário

2.1.2.1 Sobre o Produto

Inicialmente, o sistema funcionará como uma interface de comunicação entre os resultados do google form e a entrada para o pacote SOM do matlab.

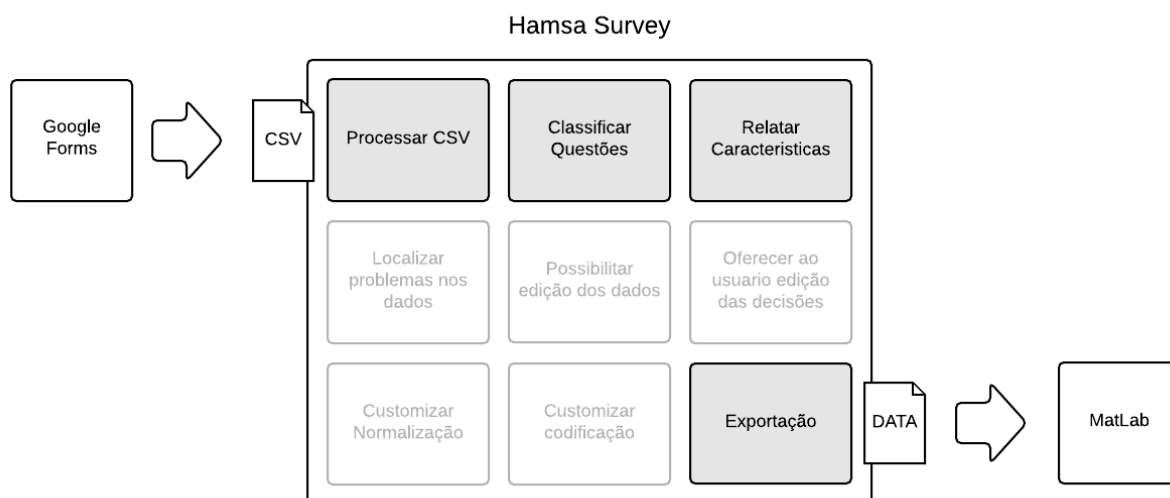


Figura 1 Macro Módulos do sistema

A imagem acima exibe em macro módulos as funcionalidades pretendidas para esta ferramenta e sua relação de interface entre os sistemas externos.

Os retângulos com fundo cinza foram implementados para esse trabalho possibilitando um produto fechado e funcional. Enquanto os retângulos rebaixados indicam funcionalidades a serem implementadas em uma próxima iteração.

Escolheu-se postergar essas etapas pois as decisões tomadas pelo usuário impactam diretamente na qualidade do resultado final, não havendo, portanto, tempo suficiente para tratar todas possíveis situações decorrentes.

2.1.2.2 Sobre o usuário

O usuário do Hamsa Survey é o usuário comum que não domina tecnologias, não é capaz de criar as próprias ferramentas ou manipular dados complexos além de como eles são apresentados. Ou seja, para que os dados façam sentido para esse usuário eles precisam estar destrinchados.

Caso o software consiga ser usado por esse usuário ideal sem dúvida poderá ser usado por outros mais aptos.

2.1.3 Ambiente operacional

O sistema será desenvolvido para Windows podendo ser portado para Linux ou OSX. Funcionará localmente não havendo necessidade de servidor ou banco de dados.

Os dados processados serão extraídos do arquivo de extensão csv advindo do google forms e sua saída funcionará como entrada para o pacote SOM toolbox do matlab.

2.1.4 Restrições de projeto e implementação

2.1.4.1 Linguagem

2.1.4.1.1 Python 3

2.1.4.2 Bibliotecas

2.1.4.2.1 Pandas

Biblioteca de código aberto para manipulação de dados.

<https://pandas.pydata.org/>

2.1.4.2.2 Numpy

Biblioteca de código aberto para computação numérica. Também utiliza estrutura de dados na forma de matrizes que é base para algumas estruturas de dados da Pandas.

<https://numpy.org/>

2.1.4.2.3 Scikitlearn

Biblioteca de código aberto com vários algoritmos de machine learning implementados. Possivelmente será usada em uma iteração futura desse projeto.

<https://scikit-learn.org/stable/>

2.1.4.3 GUI

2.1.4.3.1 PyQt

Binding de python para Qt, framework multiplataforma, que possui uma rica e bem documentada biblioteca de módulos para as mais diversas situações

<https://pypi.org/project/PyQt5/>

2.1.4.4 IDE

2.1.4.4.1 MatLab

Combina um sistema interativo cujo elemento básico de informação é uma matriz com um ambiente de desenvolvimento e linguagem de programação própria homônima ao software em questão.

<https://www.mathworks.com/products/matlab.html>

2.1.4.4.2 QtDesigner

Ferramenta para projetar interfaces complexas que podem ser importadas utilizando o PyQt

<https://build-system.fman.io/qt-designer-download>

2.1.4.5 Ferramentas

2.1.4.5.1 Som – ToolBox 2.1

Toolbox do matlab que implementa algoritmos de mapas auto-organizáveis.

<https://github.com/ilarinieminen/SOM-Toolbox>

2.1.4.5.2 Sphinx

Ferramenta de documentação automática que exporta o manual de uso para html, pdf e outros.

<https://www.sphinx-doc.org/en/master/>

Esse sistema funciona como uma interface entre o google form e o SOM toolbox do matlab. Portanto, necessariamente, o usuário precisa ter um conta no google, para produzir os formulários, e o matlab instalado com o SOM toolbox para processar os dados obtidos.

Ademais, python 3 foi escolhido como linguagem não só devido sua versatilidade como também por possuir um vasto conjunto de bibliotecas sendo, muitas delas, voltadas para tratamento de dados.

2.1.5 Documentação para usuários

A ferramenta Hamsa deverá ser um aplicativo fechado de forma que o usuário final não precise ter nenhum conhecimento sobre o funcionamento interno de seus módulos. Contudo, ao longo de seu desenvolvimento usuários mais experientes já poderão utilizá-lo mesmo antes da implementação da interface gráfica. Para isso, um manual será gerado automaticamente pela ferramenta Sphinx.

2.1.6 Hipóteses e dependências

Em iterações futuras, assume-se que o scikit learn seja capaz de substituir o matLab e que o PyQt seja capaz de fazer visualizações interativas mais complexas a partir dos mapas gerados. Caso contrário, os algoritmos utilizados no matLab precisarão ser reimplementados ou importados de implementações com licença aberta.

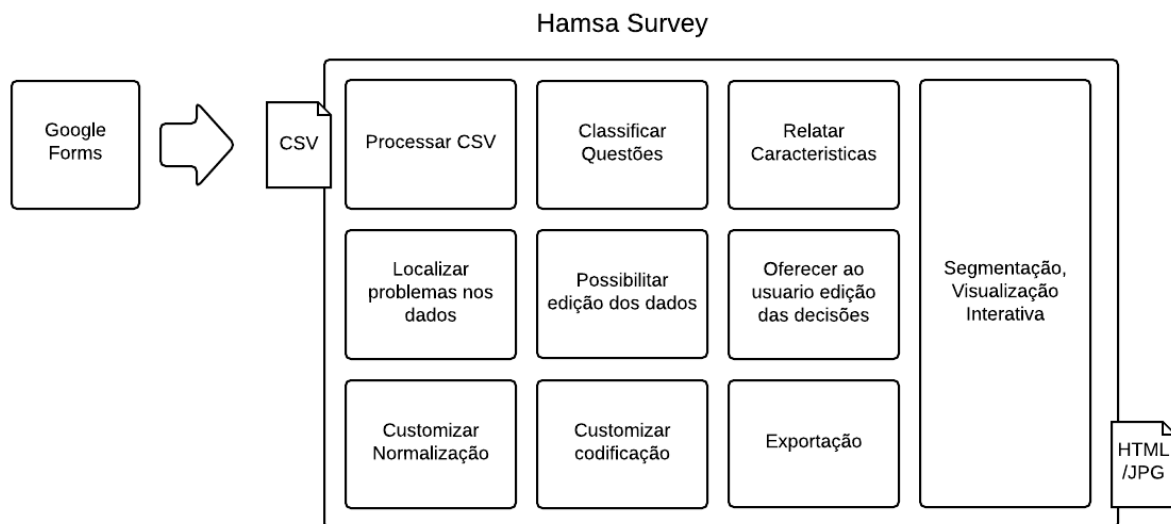
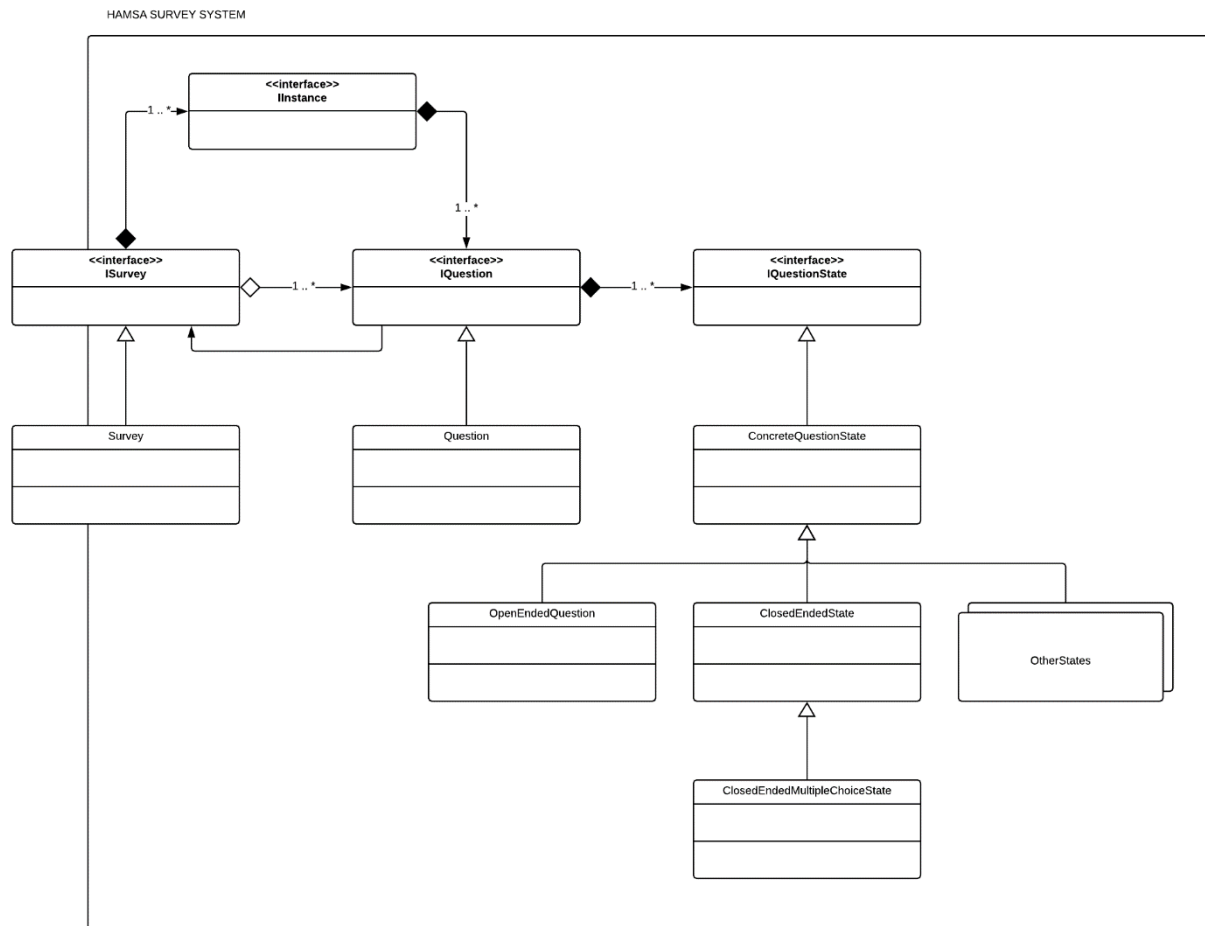


Figura 2 Possível diagrama de macro módulos para futuras iterações

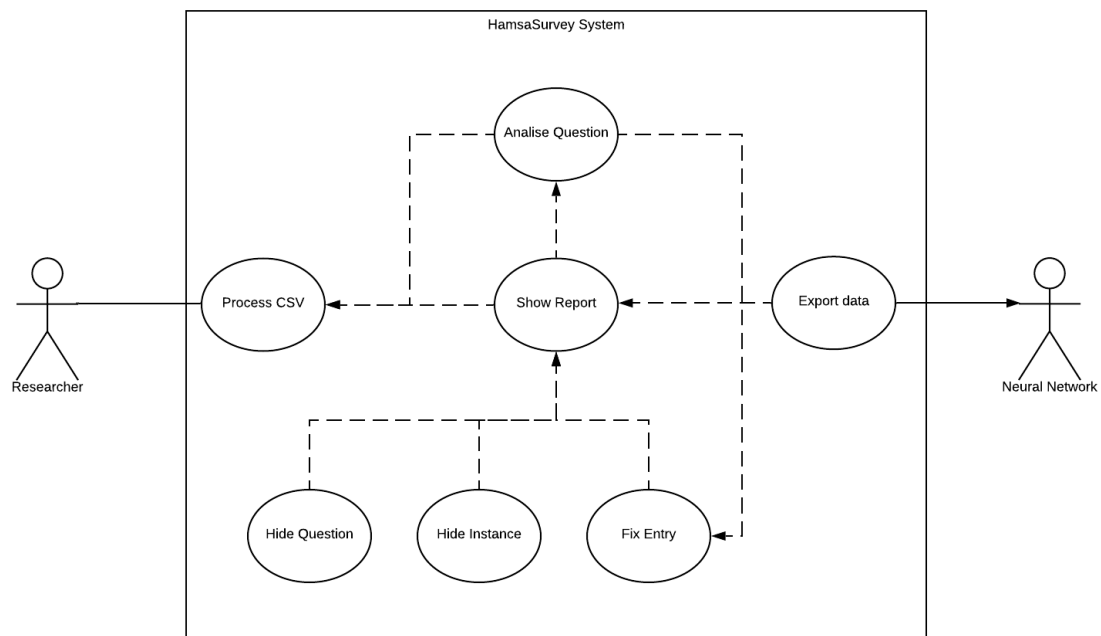
2.2 Característica do sistema

2.2.1 Descrição

Hamsa Survey é um sistema para a limpeza e preparação supervisionada dos resultados de Google Forms de forma que possam ser analisados utilizando o SOM-ToolBox do MatLab



2.2.2 Casos de uso



2.2.3 Requisitos funcionais

2.2.3.1 [RF01]- O sistema deve processar a planilha resultante do google form

A biblioteca Pandas será usada para abrir ler o arquivo CSV e sua estrutura de DataFrame funcionará como base de dados para os demais processamentos

2.2.3.2 [RF02]- O sistema deve identificar quantas e quais tipos de perguntas e respostas

Em alto nível as perguntas podem ter respostas do tipo Open-ended e Closed-ended, sendo que as Closed-ended podem ainda ter subcategorias. Essa classificação se faz necessária pois cada tipo de pergunta demanda um processamento diferenciado. O sistema, então, deverá entender qual é tipo de questão para orientar as escolhas do usuário.

2.2.3.3 [RF03]- O sistema deve verificar se as respostas se adequam ao tipo de resposta esperado

Classificada a pergunta e consequentemente o tipo de resposta esperado, o sistema deve verificar se as respostas condizem com o padrão definido.

2.2.3.4 [RF04]- O sistema deve localizar as instâncias problemáticas (respostas ausentes)

É comum ao domínio entender a não resposta como uma informação relevante. Porém, dependendo da frequência, isso pode invalidar não só a instância, mas também questão. Tais casos deverão ser marcadas para avaliação posterior.

2.2.3.5 [RF05]- O sistema deve exibir relatório

Após as etapas de avaliação o sistema deve oferecer ao usuário um relatório mostrando as características do questionário em análise e caso ocorram, uma lista dos problemas a solucionar.

2.2.3.6 [RF06]- O sistema deve oferecer a opção para ajuste manual ou a retirada da instância

As instâncias com problema serão revisitadas oferecendo a opção de ajustar manualmente ou remover.

2.2.3.7 [RF07]- O sistema deve oferecer a possibilidade de exclusão de colunas (perguntas)

TBD

2.2.3.8 [RF08]- O sistema deve mostrar a distribuição dos dados no espaço (scatter plot) e oferecer a possibilidade da criação de categorias quando necessário.

TBD

2.2.3.9 [RF09]- O sistema deve oferecer ao usuário a escolha do tipo de codificação disponível para cada questão

TBD

2.2.3.10 [RF10]- O sistema deve exportar os dados compatíveis para funcionar como entrada na rede neural

Os dados selecionados para exportação devem ser organizados no padrão exigido pelo SOM-Toolbox do Matlab e salvos no formato *.data

2.2.4 Requisitos de dados

Ao longo do fluxo de dados esses serão trabalhados de diferentes formas. Essas podem ser agrupadas em 3 etapas: Entrada, processo, saída. Ademais, os dados serão analisados e um relatório sobre suas características será exibido.

2.2.4.1 Logical data model

2.2.4.1.1 Entrada

Os dados de entrada são originados do google forms podendo ser exportado na forma de um arquivo CSV. Ele contém as X perguntas dispostas na forma de colunas e as Y instâncias de respostas dispostas em linhas.

Pergunta 1 ?	Pergunta 2 ?	...	Pergunta X?
		...	
		...	
	.		
	.		
	.		
		...	

É recomendada a utilização de um separador diferente da vírgula na exportação do arquivo CSV pois como o domínio pode exigir campos dissertativos, caso o entrevistado utilize vírgula em sua resposta o parser processará o texto equivocadamente.

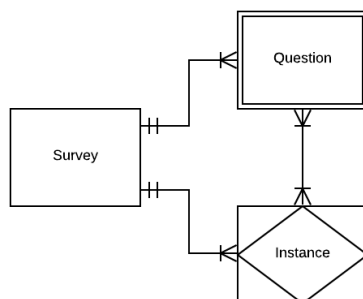
Sugere-se a utilização de “;”

2.2.4.1.2 Processo

Durante o processo haverá dois tipos de alteração a ser registrada em uma estrutura de dados persistente: Uma é a alteração do dado em si e a outra é referente a como o dado será utilizado.

As alterações do primeiro tipo, por serem os ajustes nos dados do formulário, serão salvas no próprio DataFrame da Pandas. Assim, poderão ser exportadas e importadas para vários formatos de forma barata.

As alterações do segundo tipo serão referentes a classificação dos dados, como/se serão exportados para rede neural. Demandando, portanto, a criação de uma estrutura de dados própria que seguirá diagrama de entidades-relacionamento a seguir.



2.2.4.1.3 Saída

Os dados de saída serão do formato *.data para importação no matLab. As perguntas serão substituídas por labels significativas que remetam a questão e as respostas serão trocadas pelos dados normalizados e codificados.

X				
#n	label_1	label_2	...	label_X
			...	
			...	
		.		
		.		
			...	

2.2.4.2 Data Dictionary

A estrutura de dados persistente provavelmente será uma simplificação do modelo de classes registrando o valor de seus atributos (incluindo os ponteiros para o DataFrame) de forma a ser recuperada com baixo custo.

2.2.4.3 Reports

O relatório deverá ser um resumo das características do questionário especificando os tipos de questão, tipos de resposta, instâncias com problema.

2.2.4.4 Data Aquisition, Integrity, Retention and Disposal

A estrutura mencionada na seção 2.2.4.2 será exportada no formato json de forma a ser recuperada em outra seção do aplicativo caso seja necessário.

2.3 Requisitos de interfaces externas

2.3.1 Interfaces do usuário

A GUI do Qt será utilizada para compor e customizar as janelas modais que exibem as informações do questionário em três níveis (Questionário, Questão, Instância) e uma tela de inspeção que permita a navegação entre perguntas e respostas.

2.3.1.1 Questionário

Exibição da entidade Survey. Apresenta uma visão geral do questionário mostrando as questões, a qual tipo cada uma pertence e suas labels. Além disso, indica as instâncias possivelmente problemáticas e questões que demandem um ajuste manual.



The screenshot shows a window titled "Questionário" with a text area at the top containing Lorem Ipsum text. Below it is a table with three columns: "label", "type", and "question". The table has 10 rows. At the bottom right, there is an "export" button.

label	type	question

2.3.1.2 Questão

Essa tela exibe apenas uma questão, sua label, sua classificação e as respostas de todas as instâncias. Nela é possível alterar cada uma das informações exibidas.



The screenshot shows a window titled "Questão" with two input fields at the top: "label" and "type". Below them is a text area containing Lorem Ipsum text. At the bottom, there is a table with 10 rows and 2 columns.

label	type

2.3.1.3 Instância

Essa tela exibe apenas uma instância e suas respostas. Além disso, é possível ver os labels de cada pergunta de forma a saber a qual pergunta aquela resposta se refere. As respostas serão exibidas de forma diferente em função do tipo de questão. TBD

de resposta (ensino médio, superior e pós-graduação) cada uma será substituída por um valor (0, 1, 2, respectivamente). Então, os valores substituídos serão divididos pelo maior valor (no caso 2) de forma que todos as respostas fiquem dentro do range 0-1.

O arquivo então é composto substituindo as perguntas por labels e adicionando os indicadores do número de perguntas (no canto superior esquerdo “3” e “#n”).

2.4 Outros requisitos não funcionais

2.4.1 Usabilidade

O software deve ser fácil de utilizar mesmo para pessoas não acostumadas com limpeza ou manipulação de dados.

2.4.2 Consistência

A interface deve ser consistente com o sistema operacional de forma que o usuário saiba o que fazer sem precisar fazer testes.

2.4.3 Extensibilidade

O sistema deve ser construído modularmente de forma que possa ser estendido incorporando o sistema de redes neurais em iterações futuras.

2.4.4 Recuperabilidade

Caso o sistema caia ele deve retornar para o estado anterior.

3 Arquitetura de Software

3.1 Introdução

Esta seção oferece uma visão em alto nível do projeto de arquitetura de software da ferramenta Hamsa Survey. O objetivo desse artefato não só é comunicar decisões significativas em relação aos componentes utilizados para atender aos requisitos funcionais e não funcionais (especificados no documento *especificação dos requisitos de sistema*) mas também, iterativamente, ampliar o entendimento do próprio projeto durante o planejamento.

3.2 Detalhamento técnico dos requisitos não-funcionais

3.3 Usabilidade

3.3.1 [RNF1] utilizar hints

Sempre que o texto for maior que o campo, exibir o texto via hint

3.3.2 [RNF2] Navegação para edição

Os campos das tabelas sempre podem ser editados mediante duplo clique.

3.3.3 [RNF3] Navegação para inspeção

O duplo clique no header da linha ou da coluna abre a respectiva tela modal. Nessa, a linha / coluna em questão será exibida juntamente com as informações relacionadas.

3.4 Consistência

3.4.1 [RNF4] Interface

A UI do qt naturalmente oferece uma interface coerente com o sistema operacional.

3.5 Extensibilidade

3.5.1 [RNF5] Modularidade do tipo de questão

Novos tipos de questão devem poder ser inseridos utilizando a máquina de estados

3.5.2 [RNF6] Modularidade do tipo de codificação

Novos métodos de codificação devem poder ser inseridos, provavelmente utilizando decorator pattern

3.5.3 [RNF7] Modularidade do tipo de normalização

Novos métodos de normalização devem poder ser inseridos, provavelmente utilizando decorator pattern

3.6 Recuperabilidade

3.6.1 [RNF8] Dados persistentes

As decisões do usuário devem ser registradas persistentemente de forma que a sessão possa ser retomada que sejam perdidas.

3.7 Performance

Nessa primeira iteração a performance não é uma questão uma vez que o processamento é de limpeza é de baixo custo computacional. Esse requisito será revisto quando da incorporação da etapa de segmentação.

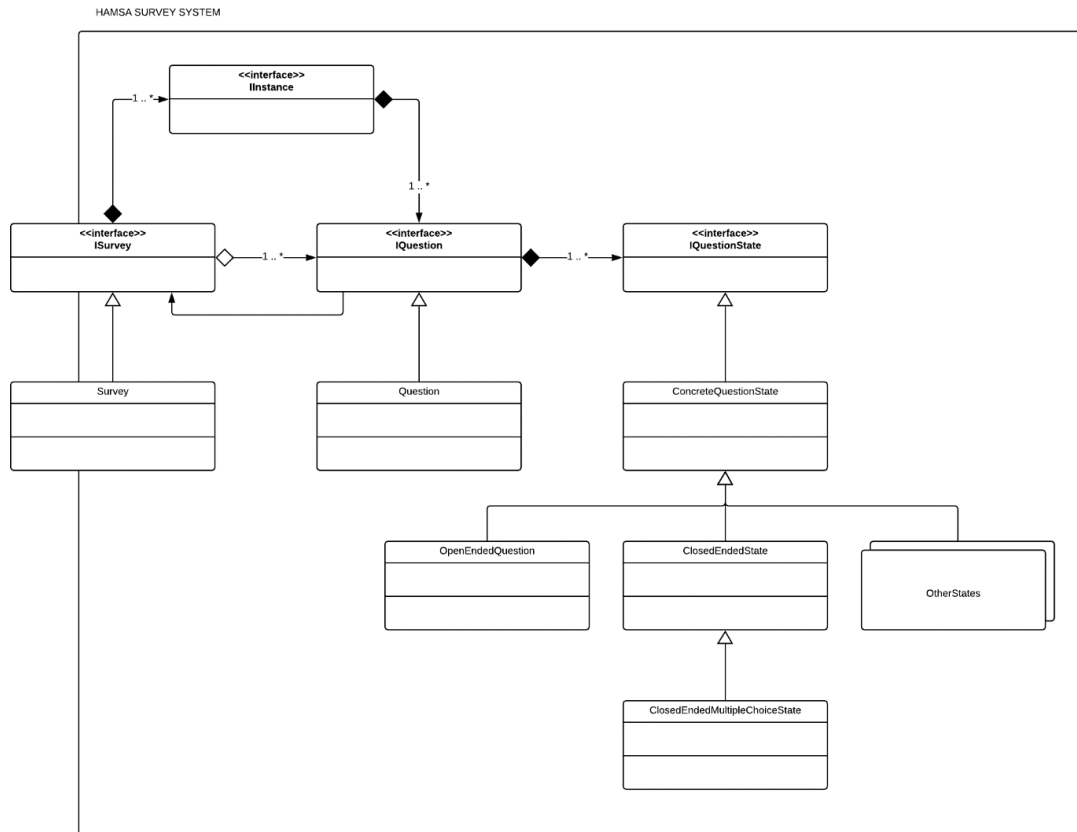
3.8 Componentes de Solução

O sistema foi decomposto em componentes com o objetivo de entender melhor os problemas, isolá-los e criar soluções reaproveitáveis e expansíveis. Para isso utilizou-se alguns padrões de projeto sugeridos no livro Design Pattern (Gamma, Erich and Helm, Richard and Johnson, Ralph and Vlissides, John, 1995).

Os padrões utilizados foram:

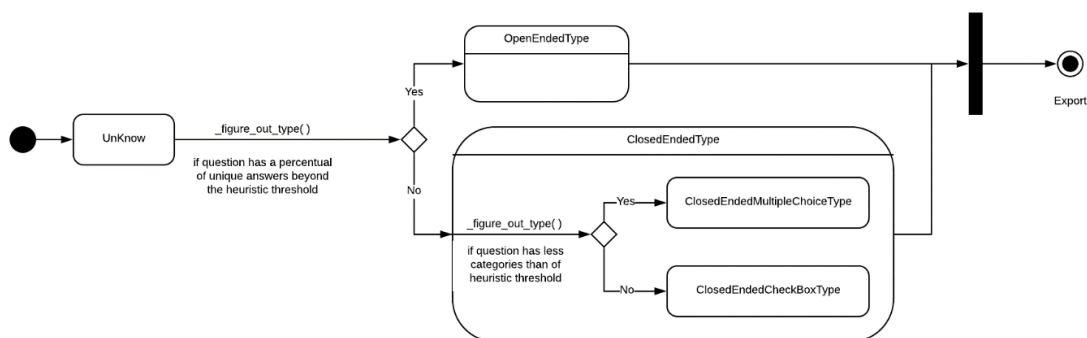
3.8.1 Façade

A classe Survey foi criada atuando como uma interface entre o sistema e a GUI. Através dela é possível facilmente fazer solicitações de respostas, questões, exportação, codificação, normalização entre outras.



3.8.2 Factory

A classe questão funciona utilizando o padrão Factory. A partir de características específicas das respostas ele decide qual classe de tipo irá instanciar.



3.8.3 State

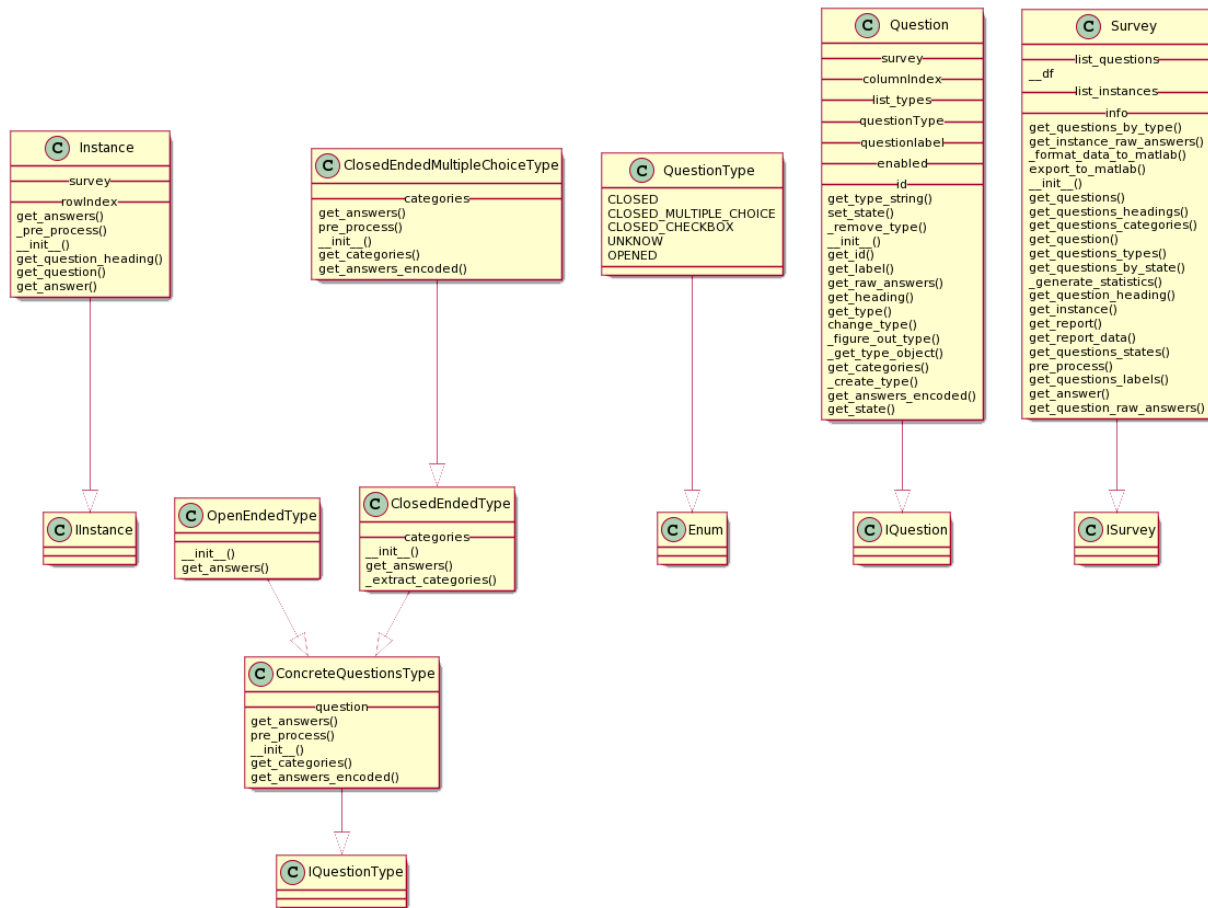
As questões utilizam os tipos como estado, conforme sugerido no padrão State. Dessa forma, ao modificar o tipo da questão modifica-se também como o dado é preparado durante as etapas de codificação e normalização.

Eventualmente, as features das questões precisarão ser extraídas dos textos, como é o caso das `OpenEndedQuestions`. O padrão escolhido já resolve essa situação.

3.8.4 Decorator

Em futuras iterações pretende-se utilizar esse padrão para encapsular tratamentos dos dados nas etapas de Codificação e normalização.

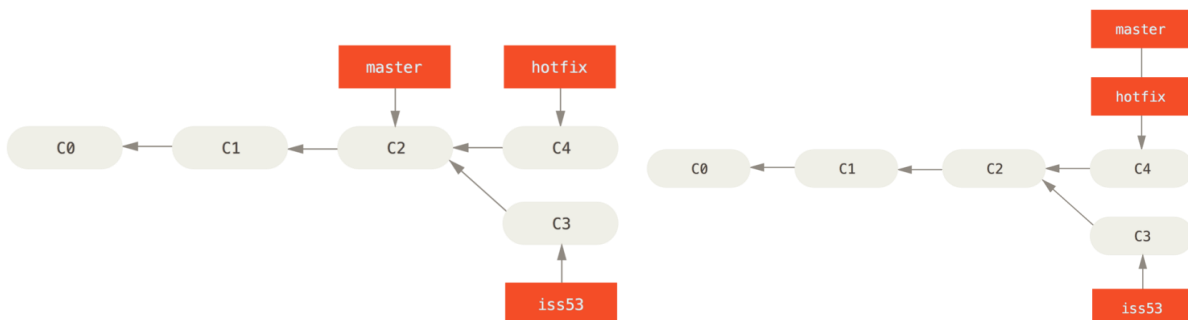
3.9 Visão de implantação



3.10 Controle de versão

O Git foi escolhido como ferramenta para controle de versão. Nele é possível dividir a linha de desenvolvimento em branches e commits. O primeiro funciona como ramificações desta linha e o segundo como *milestones*.

Assim, foi possível criar *branches* para testes, correções, desenvolvimento de tarefas específicas entre outros. Deste modo, os estados funcionais de desenvolvimento são salvos e podem ser fundidos com os novos tão logo sejam concluídos conforme ilustrado na figura abaixo.



Os *Branches* e *commits* foram criados seguindo os critérios apresentados nas subseções a seguir.

3.11 Branches

- Existem dois *branches* principais: Master e Devel.
- Os *branches* são criadas utilizando o código do requisito funcional (descrito na especificação de requisitos do sistema) a ser implementado. Ex. RF01, RF03
- *Branches* com o sufixo **_try** são criadas para implementações de teste, não necessariamente utilizadas.
- Sempre que uma tarefa é concluída é feito o *merge* com a *branch* Devel.
- Ao fim da primeira iteração, será feito *merge* da *branch* Devel com o Master.
- Outros *branches* são inseridos de acordo com a necessidade. Busca-se manter os nomes auto-explicativos.

3.12 Commits

- *Commits* são feitos para registrar estados funcionais, backup, transferências de arquivos entre computadores, fim de tarefas ou preparação para alguma etapa em particular.
- A descrição de todo *commit* é precedida por “[]”.
- Dentro do colchete é registrado o número do *commit* naquele *branch* e o código do requisito que esta sendo trabalhado.
- Sempre que houver OK logo após o colchete trata-se de um *commit* funcional ou encerramento de alguma tarefa.

A listagem a seguir ilustra a situação:

```
PS C:\Users\Bruno\Documents\dev\hamsa_survey> git log --oneline --all --graph
* be4a32d (HEAD -> dev, origin/interface_try, interface_try) [2] OK
* 58c7a64 [1] export working import into matlab working
* d0958d2 [0]Interface - report screen almost working
* c879be8 (origin/refactor_pattern, refactor_pattern) [5] backup
* b4a8a2a [4]methods commented
* 82c18f0 [3] OK - state pattern working
* 889b0c7 [2] quick backup
* bd8a4da [1] state machine to change question type
* 2086336 [0] refactoring design pattern
* c5215e7 (RF05, RF02_try) [RF02 - 05] OK WORKING RF02 Sistem can figure out questions type
| * c4e10b1 (refs/stash) WIP on fixGittree: 43d23ae [RF02 - 04] fixing problems with git
|/|
| * 27f5148 index on fixGittree: 43d23ae [RF02 - 04] fixing problems with git
|/|
* 43d23ae (origin/fixGittree, fixGittree) [RF02 - 04] fixing problems with git
* 2ad89e6 (origin/hotfix, hotfix) [RF02- 03] module structure fixed
* 43271da [RF_02 - 02] backing up module structure
* 44dedb6 RF02- 02 chenging module structure again
* f8e4188 (origin/RF02_try) [RF02 - 02] saving backup to change modules structure
* 3a07619 [RF02 - 01] solving problem with modules inheretance
* 712ea4b [RF02 - 00] figuring out which type questions are.
* d261f90 (origin/dev) [RF01 - 01] OK - opening CSV file. Token to parse can be changed via parameter
* 40f60a4 [RF01 - 00] token as param to preprocess the csv file
* 3328994 (origin/autodocconfig, autodocconfig) interface mockup added
* 3c471b3 first interface test
* d423e43 Auto documentation configured using Sphinx
* edd6bcf (origin/master, origin/HEAD, master) Initial commit
```

4 Testes

4.1 Verificação e Validação

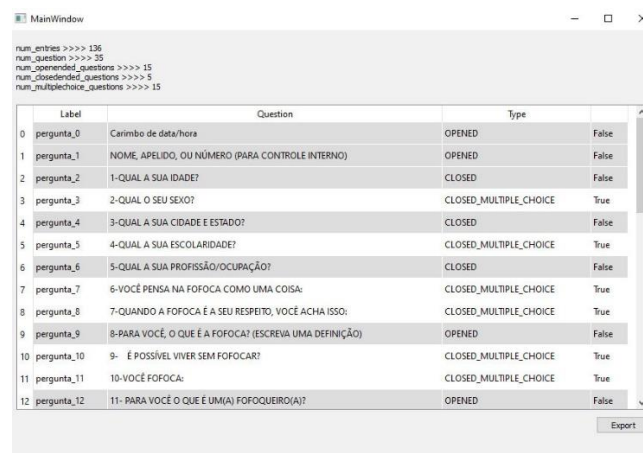
Os documentos de especificação de requisitos de sistema e arquitetura de software serão utilizados como referência nessas etapas. A detecção de defeitos será Ad-Hoc.

4.2 Situação Atual

Apenas a tela de relatório foi implementada permitindo ao usuário ver como cada pergunta foi caracterizada.

O sistema classifica as questões entre OpenEnded, ClosedEnded, ClosedMultipleChoice. Sendo apenas o último tipo exportado para a rede neural. Todas as demais são desabilitadas.

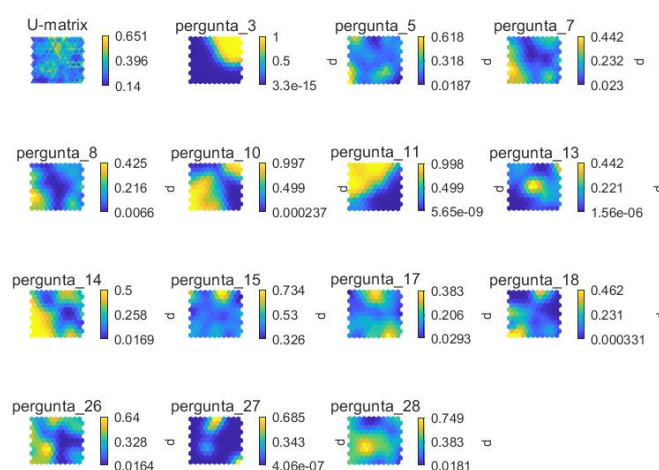
Apesar do botão exportar ainda não estar funcionando o sistema já está exportando automaticamente para o arquivo “matlaba.data” na raiz do projeto.



	Label	Question	Type	
0	pergunta_0	Carimbo de data/hora	OPENED	False
1	pergunta_1	NOME, APELIDO, OU NÚMERO (PARA CONTROLE INTERNO)	OPENED	False
2	pergunta_2	1-QUAL A SUA IDADE?	CLOSED	False
3	pergunta_3	2-QUAL O SEU SEXO?	CLOSED_MULTIPLE_CHOICE	True
4	pergunta_4	3-QUAL A SUA CIDADE E ESTADO?	CLOSED	False
5	pergunta_5	4-QUAL A SUA ESCOLARIDADE?	CLOSED_MULTIPLE_CHOICE	True
6	pergunta_6	5-QUAL A SUA PROFISSÃO/Ocupação?	CLOSED	False
7	pergunta_7	6-VOCE PENSA NA FOFOCA COMO UMA COISA?	CLOSED_MULTIPLE_CHOICE	True
8	pergunta_8	7-QUANDO A FOFOCA É A SEU RESPEITO, VOCE ACHA ISSO?	CLOSED_MULTIPLE_CHOICE	True
9	pergunta_9	8-PARA VOCE, O QUE É A FOFOCA? (ESCREVA UMA DEFINIÇÃO)	OPENED	False
10	pergunta_10	9- É POSSIVEL VIVER SEM FOFOCAR?	CLOSED_MULTIPLE_CHOICE	True
11	pergunta_11	10-VOCE FOFOCA:	CLOSED_MULTIPLE_CHOICE	True
12	pergunta_12	11- PARA VOCE O QUE É UM(A) FOFOQUEIRO(A)?	OPENED	False

Figura 3 Tela de Relatório

A exportação para rede neural funciona corretamente. Os dados foram importados no matlab gerando a seguinte segmentação:



SOM 28-Aug-2020

Figura 4 Segmentação gerada pelo matlab com base dataset proposto.

4.3 Requisitos Funcionais

RF01	O sistema deve processar a planilha resultante do google form	OK
RF02	O sistema deve identificar quantas e quais tipos de perguntas e respostas	OK
RF03	O sistema deve verificar se as respostas se adequam ao tipo de resposta esperado	
RF04	O sistema deve localizar as instâncias problemáticas (respostas ausentes)	
RF05	O sistema deve exibir relatório	OK
RF06	O sistema deve oferecer a opção para ajuste manual ou a retirada da instância	
RF07	O sistema deve oferecer a possibilidade de exclusão de colunas (perguntas)	
RF08	O sistema deve mostrar a distribuição dos dados no espaço (scatter plot) e oferecer a possibilidade da criação de categorias quando necessário.	
RF09	O sistema deve oferecer ao usuário a escolha do tipo de codificação disponível para cada questão	
RF10	O sistema deve exportar os dados compatíveis para funcionar como entrada na rede neural	OK

4.4 Requisitos Não Funcionais

RNF1	Utilizar hints	OK
RNF2	Navegação para edição	
RNF3	Navegação para inspeção	
RNF4	Interface	OK
RNF5	Modularidade do tipo de questão	OK
RNF6	Modularidade do tipo de codificação	
RNF7	Modularidade do tipo de normalização	
RNF8	Dados persistentes	

5 References

Gamma, Erich and Helm, Richard and Johnson, Ralph and Vlissides, John. (1995). *Design Patterns: Elements of Reusable Object-Oriented Software*. USA: Addison-Wesley Longman Publishing Co., Inc.

McConnell, S. (2004). *Code Complete, Second Edition*. Acesso em 31 de 8 de 2020, disponível em <http://gbv.de/dms/ilmenau/toc/38547900x.pdf>

Pressman, R. (2010). *Software Engineering: A Practitioner's Approach*. McGraw Hill. Acesso em 31 de 8 de 2020

Sommerville, I. (2011). *Software engineering*. Pearson Education. Acesso em 31 de 8 de 2020, disponível em <http://iansommerville.com/software-engineering-book/files/2015/08/Ch-12-Safety-Engineering.pdf>