Faculdade de Design, Tecnologia e Comunicação
Universidade Europeia

INTELLIGENT AWARENESS

1. Project Description

Project overview

The objective of this project is to tackle a very specific but threatening problem in today's society that is unwanted loneliness, it mainly affects the quality of life of our senior citizens (65+ years old, 2,2Million in Portugal [2019 Pordata]), it affects their mental health, social interactions, emotional and even physical health. In today's age where the whole world has been affected by a major pandemic it has become more and more urgent to find ways to combat the impact it has. The social and physical isolation come with big repercussions to those that live alone.

Main project goals

The goal is to develop an algorithm that identifies whether a specific person is already or going towards this process of unwanted loneliness by measuring that person's churn rate (metric for withdrawal) and that person's activeness. For that we will build various types of communities so everyone could be included and join a community of their preference. Inside those communities we will have a community monitor, an expert, that will assign tasks for the targets (community members) and also monitors the communities' performances according to the task's completions.

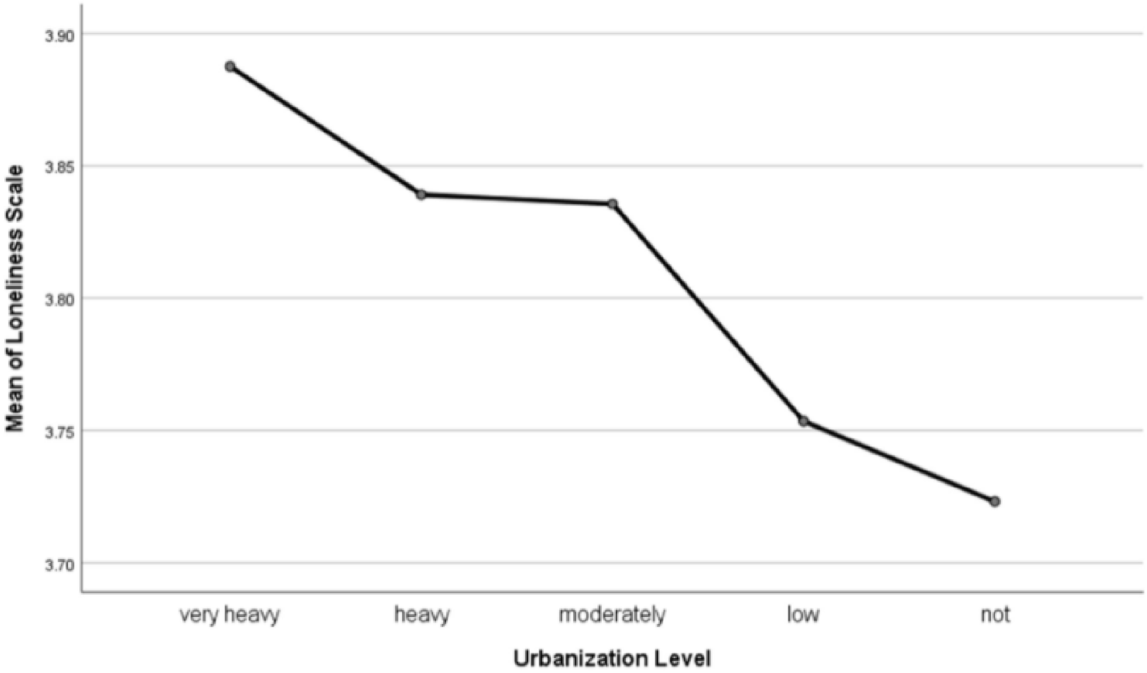


Figure 1. Plot of lonely scale scores by urbanization levels [6]

Survey of existing algorithms

A research of existing algorithms was done at the start, we tackled and focused on finding algorithms that either predicted mental health status or churn rate of a specific person.

”The author used AUC to measure the performance of the algorithms. The AUC values were 99.10%, 99.55% and 99.70% for Bayes Networks, Neural networks and support vector machine, respectively” [1]. Although the tree-based algorithms had the best results (Decision tree, Random Forest, GBM tree algorithm, and XGBOOST algorithm) when the problem scaled to bigger dimensions (5.23 million Chinese telecom users) only the neural network could sustain the scalability problem, maintaining above 91% accuracy.

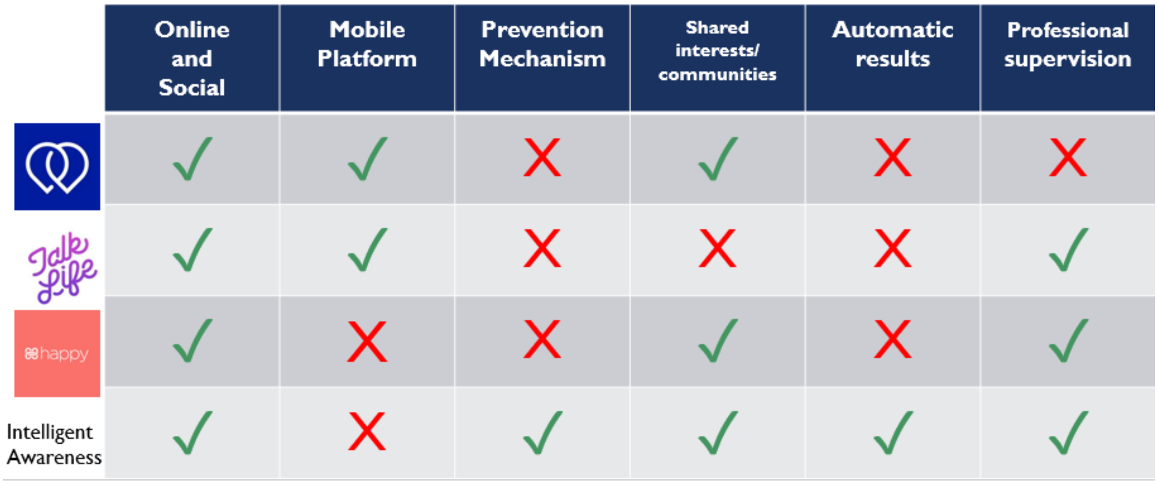
Benchmarking analysis

A benchmarking analysis of existing applications was made in order to find if any important change in our product would be made according to these applications strong and weak features. We found out that Panion, TalkLife and Happy had the same objective as ours.

A brief description of the apps:

- Panion (100K+ downloads Feb. 2021, 3.5 stars on the google app store) - Permits their users to find other users that have the same interests, allowing them to later communicate between them.
- TalkLife (1M+ downloads Feb. 2021, 4.2 stats on the google app store) - Permits their users to find a random person to talk to about their struggles and/or interests.
- Happy (website) – Similar to the last two, also allows their users to find communities and to talk with random or specific people about their struggles and/or interests.

From them, we analyzed their strong points and compared to ours:



	Online and Social	Mobile Platform	Prevention Mechanism	Shared interests/ communities	Automatic results	Professional supervision
	✓	✓	✗	✓	✗	✗
Talk Life	✓	✓	✗	✗	✗	✓
happy	✓	✗	✗	✓	✗	✓
Intelligent Awareness	✓	✗	✓	✓	✓	✓

We also analyzed their net-sentiment, a metric that weights people social media opinions on a specific product. We subtract bad opinions from the good opinions and then divide by all the opinions (good, bad, and neutral). The platform we used was Brand24:

Panion

$$\frac{16-5}{88} = 12{,}5\%$$

TalkLife

$$\frac{26-6}{52} = 38{,}4\%$$

Happy

$$\frac{24-2}{43} = 51{,}1\%$$

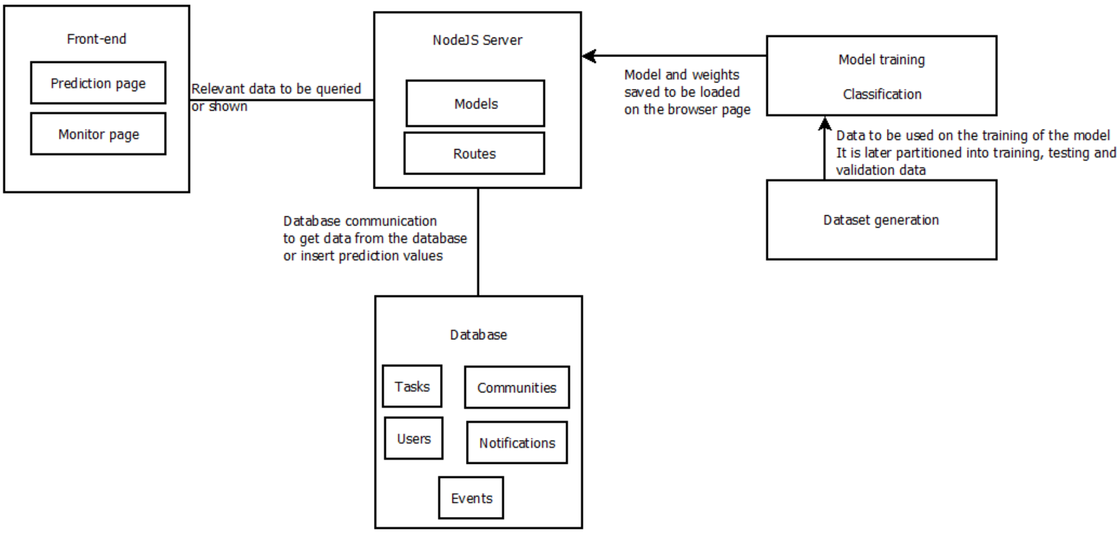
note: Happy's net sentiment is extremely high because of the immense number of applications with the same name, making it difficult for us to get the exact amount.

And from this, we conclude that our ideas differentiate enough from the concurrent applications for us to change anything dramatically. But making our product support a mobile environment would be the biggest change to be made.

2. Software Design

Software infrastructure

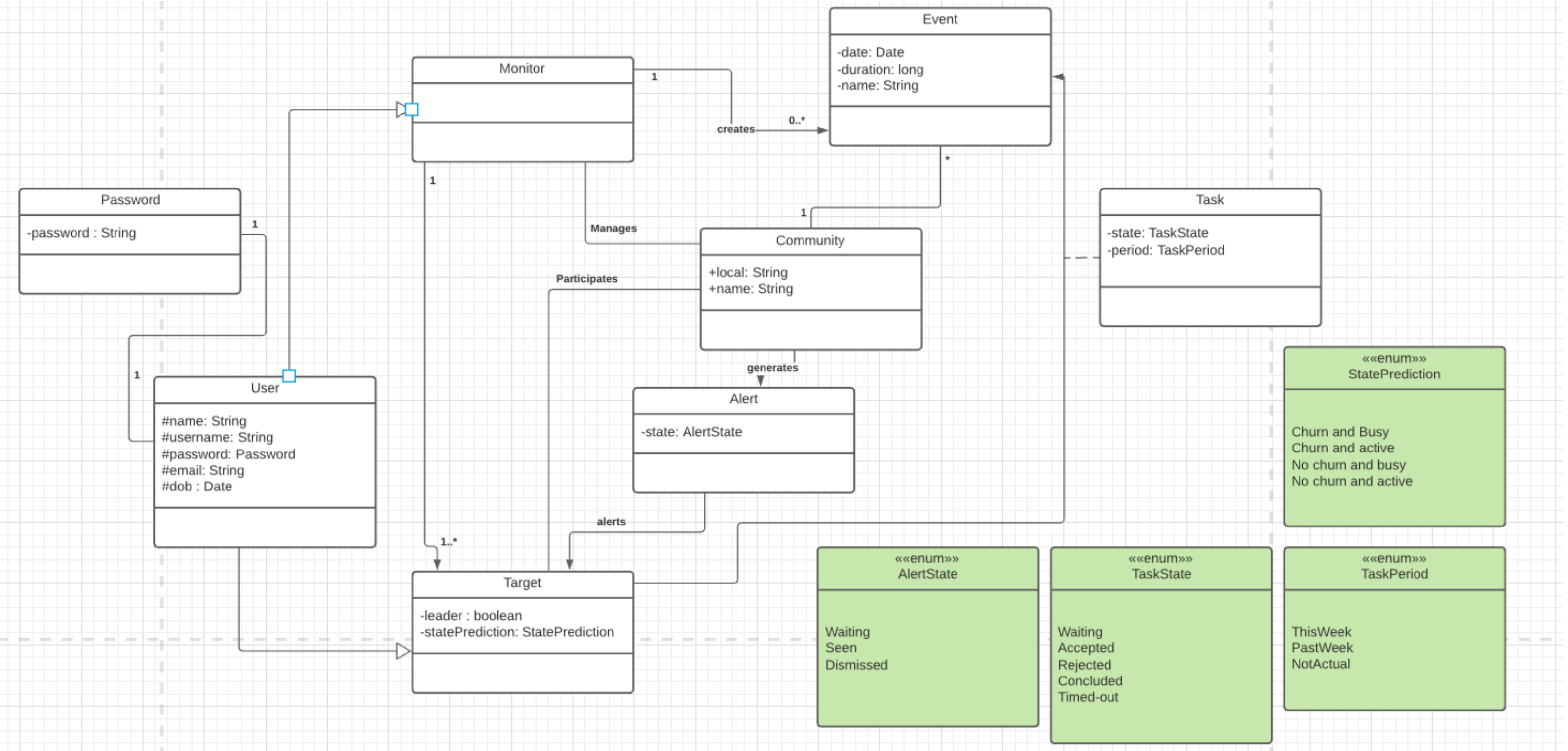
- Architecture mockup overview


Front-end
Prediction page
Monitor page
Relevant data to be queried or shown
NodeJS Server
Models
Routes
Model and weights saved to be loaded on the browser page
Model training
Classification
Data to be used on the training of the model It is later partitioned into training, testing and validation data
Dataset generation
Database communication to get data from the database or insert prediction values
Database
Tasks
Communities
Users
Notifications
Events

- Software components
    - The front-end is a webpage that will allow us to predict a specific person churn and activity rate by inserting their recent performance (and data that lets us isolate which person it is so we can get past performances from the database) and will allow a monitor to have an overview of their community's performance.
    - NodeJS server will allow us to route the frontend requests to the database.

- On the Database we store relevant data
- On the Model training/classification, we build up a model that will be saved and loaded on the frontend to then make predictions according to the browser's requests.
- On the Dataset generation we will generate data to be tested/trained, see more on the dataset section.

Information architecture



a. Users is where we will store personal data for each user, nothing more than enough for the project to function well.

b. Password is where we will store the user's unique password, we decided to create a new table specifically for it so it wouldn't come with the Users information.

c. Monitor is a User that has extra functionalities (that are not specified on the diagram). Things like having access to a dashboard of performances.

d. Community is the group of people that share the same interest, it will be moderated by the Monitor.

e. Target is a user that is the Community member, the target of the project.

f. Event is a Task that is created by the Monitor to be performed by the whole community (all the targets inside the community).

g. Task is where we link each event to the specific target.

h. The Alert table is our notification system.

i. AlertState is an enumeration that determines if the notification has been sent, seen, or dismissed.

j. TaskState is an enumeration that determines if the Task has become available, has been accepted, rejected, completed or timed-out.

k. TaskPeriod is where we control the periodicity of the task, how recent it is.

l. StatePredicition is where we store the prediction made by our model to a certain target.

Algorithms and Data Structures

Dataset

A dataset is a collection of data to be interpreted by the computer, it has to be coherent, uniform and understandable for the machine. In this case, the dataset we need is for the predictions and for the progress representation. Dealing with this part of the project was the most time-consuming part, everything depended on it, from the KPIs (Key performance indicators) we wanted to analyze to the dimension and scale the whole project could grow into. Sadly, a specific dataset for this problem probably doesn't exist, we will now explain the whole process that got us to our dataset.

Towards a dataset

Towards the dataset, we wanted a dataset that would describe profiles (each row would be a person) and wanted it to have a chronological characteristic, so we could specify who the person was and have recording of progress throughout the whole process.

After a long research, we determined that there was not a perfect dataset out there for us to use. Although we could find a similar one to then change some attributes to fit our case, we could not find one that was good enough, so we decided to generate our own synthetic dataset. Here are some of the datasets we tried to fit with our problem:

- Dataset for predicting mental situation of people in covid-19: With this dataset.
- Loneliness and Social Connections: This dataset objective was to prove that social connections are important for our mental health and well-being. However, their method was to analyze social connections across countries over time, which approached our goal of a chronological aspect but did not approach our profile characteristic.
- Isolation and Loneliness of Older People: This one tackled the same issues we wanted to, but it was private and required a formal data request which we did thankfully for Moai LABS and their Project Manager Dr. Gabriel Pestana who filled the request form but with no success or answer.
- Predict alzheimer disease sl and tf: Same idea as ours, but different prediction, which we could change their values to fit to ours, but it didn't have the chronological aspect.

And many other that we decided not to include because the reasons do not vary from these. Then it was decided a synthetic dataset creation was the way to go.

Dataset description

Since the conclusion was to make our dataset, we needed to find how we wanted to model it. Having the "profile" aspect and chronological aspect as requirements, this are the KPIs(Key performance indicators) we decided to use:

- Available tasks.
- Accepted tasks.
- Cancelled/rejected tasks.
- Completed tasks.
- How many times the person accessed the platform on average.
- Time spent on the platform on average.
- Average days to conclude the tasks.
- If the person was a Leader past week.

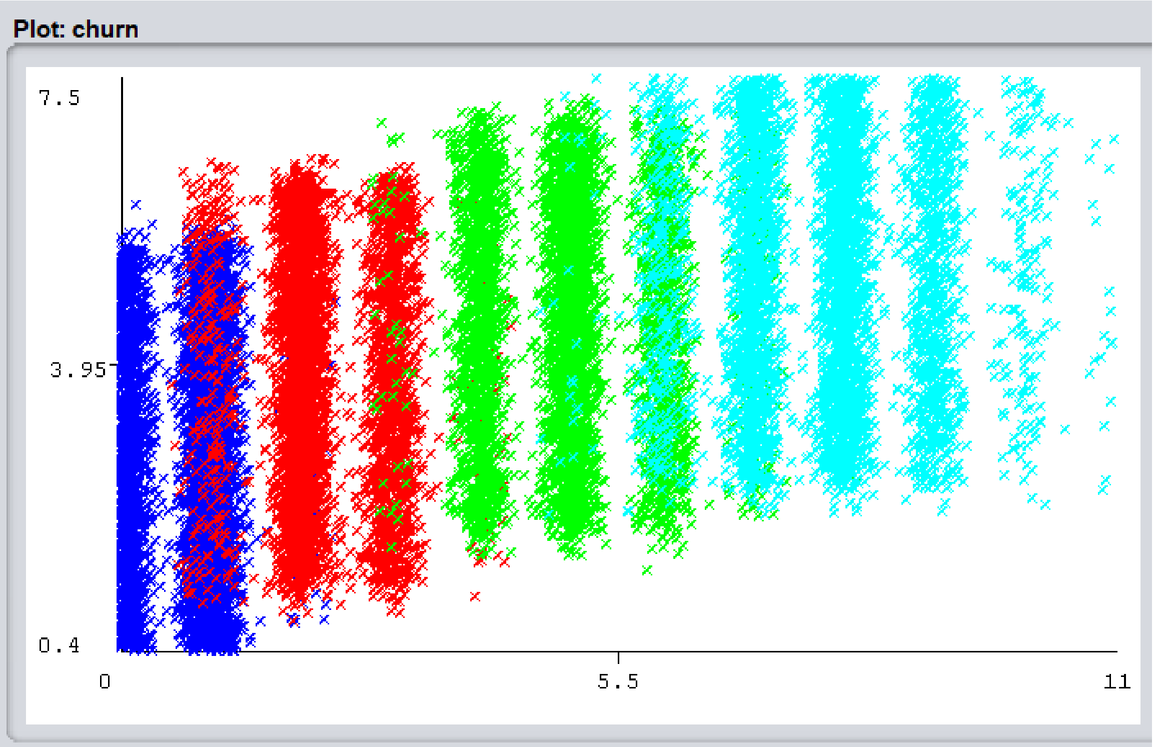
- If they completed the Leader task.

With these 9 KPIs, we duplicated them and labeled them as Previous week and This week, so we would tackle the chronological aspect, plus the label at the end makes 19 columns.

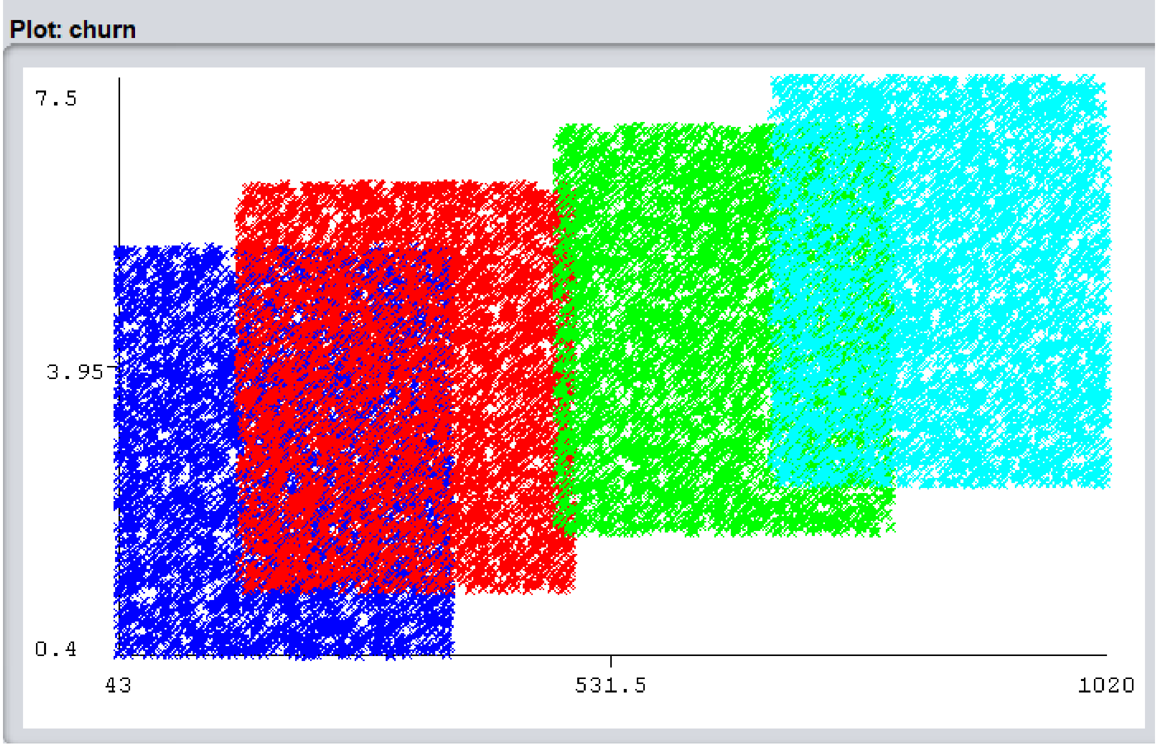
Each row would be an individual person, this is how one row would look like:

10 tasks available last week, 5 tasks accepted last week, 4 tasks cancelled last week, 1 task completed last week, 1,75 average times they accessed the platform last week, 183 average seconds per platform access last week, average 6 days to conclude a task last week, this person was not a leader last week, this person did not complete the previous week leader task, 14 tasks available last week, 11 tasks accepted last week, 6 tasks cancelled last week, 0 task completed last week, 0,65 average times they accessed the platform last week, 88 average seconds per platform access last week, average 7 days to conclude a task last week, this person was not a leader last week, this person did not complete the previous week leader task, this person label is Churn and Busy.

Here are some graphs to show the variation between the outputs:



times accessed platform by Concluded tasks



Times accessed platform by time spent online

Dataset generation

For the generation, we defined a mean for each of our 9 KPIs for each of our Profiles, so each profile would have different means on each KPIs, for example a busy person with churn wouldn't have the same mean value of tasks completion as an active and not busy person.

Churn and busy

- Accepted tasks mean: 51%
- Cancelled tasks mean: 60%
- Concluded tasks mean: 60%
- average times accessed platform: 2.4 times
- average time spent on the platform: 30 seconds
- average days to conclude a task: 5 days

Churn and active

- Accepted tasks mean: 57%
- Cancelled tasks mean: 33%
- Concluded tasks mean: 72%
- average times accessed platform: 3.2 times
- average time spent on the platform: 60 seconds
- average days to conclude a task: 3 days

No churn and busy

- Accepted tasks mean: 75%
- Cancelled tasks mean: 20%
- Concluded tasks mean: 90%
- average times accessed platform: 3.9 times
- average time spent on the platform: 130 seconds
- average days to conclude a task: 2.5 days

No churn and active

- Accepted tasks mean: 95%
- Cancelled tasks mean: 8%

Concluded tasks mean: 90%
average times accessed platform: 4.5 times
average time spent on the platform: 160 seconds
average days to conclude a task: 1.5 days

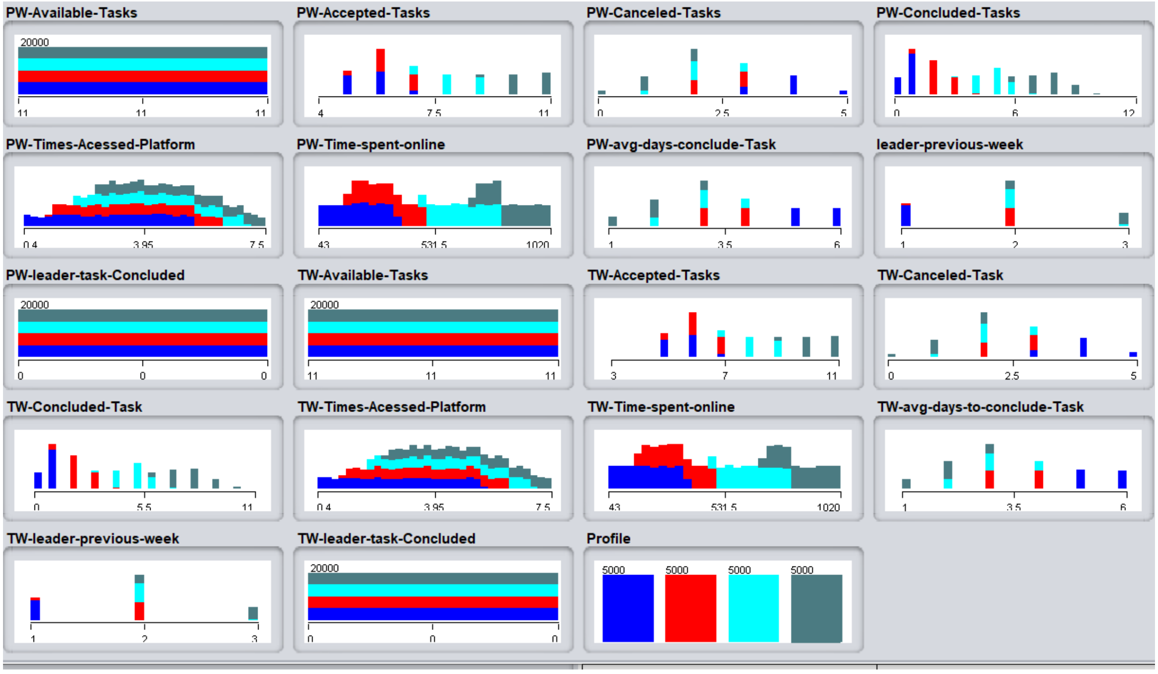
The available tasks amount for each week was the same for all the rows, and it had a mean of 11.

Now we inserted some variance so it would become unlikely for two rows to have the same stats and also so the values would overlap.

For the last two KPIs, to check if the person was a leader (someone who did very well performance-wise) last week and if they concluded their leader task we did implement and very simple gamification algorithm, depending on the percentage of available, accepted, cancelled, and completed tasks, we will attribute them with the Leader tag, and for the latter, we used the same concluded task percentage given to that specific person to determine if it was completed or not.

For a better understanding, please check the generation algorithm here: https://github.com/ElynoLamar/IntelligentAwareness/blob/master/code/public/javascripts/generateDataset.js

Below we have the distribution of our KPIs according to the profiles (output values/labels).



(1(blue)-> churn and busy, 2(red)-> churn and active, 3(cyan)-> no churn and busy, 4(grey)->no churn and active).

Algorithms for problem Solving

Deep Learning

Initially we need to upload the dataset that we need to use, for that we use the library Pandas that allows us to turn a dataset separated by commas (cst) into a DataFrame using pandas.read_csv, and we give a name for the features(18 first columns) and for the label(last column). After that we need to shuffle the dataset, it is a good practice and allows the model to be trained in a way that reduces bias, breaking the structure of the dataset, we used numpy.random.permutation() for that. After shuffling we need to index each row, re-ordering the dataset, since the dataset has been modeled into a DataFrame with pandas, we can use reset_index() for the re-ordering of the indexes.

Now we need to select and divide the dataset between training data and test data, the training data will be the partition that the model will learn from, the test data will be the partition that the model will test its capabilities. Since we're using tf.keras for the model, the model.fit() function only accepts either tensors or numpy arrays as inputs, so we had to change our data from pandas DataFrame to numpyarray.

We then configure the layers in the model, the first layer (input layer) has to collect 18 inputs so the input_shape has to be (18,), we chose ReLU for the activation function of the first layer, since it is one of the fastest growing function just like Sigmoid, but since we use a non binary output, ReLU has a better fit.

Since we have a non-binary (multiclass) number of outputs, we need to use a function that helps us determine the probability of each class, and that is what Softmax function does, it divides the exponential of one input by the sum of the rest of the inputs, and that normalizes the probability distribution between the multiple classes, not allowing those probabilities to be negative or above 1. We used softmax on our last layer, the output layer which has 4 nodes, one for each output.

We need a loss function that determines through each epoch how far are we from what we want, good results, and for that we use Cross-entropy as our loss function, it calculates the difference between two probability distributions, one is the overall training inputs, the other is the desired output. Note that we need to use categorical cross entropy for multi class outputs.

To get better results, we need something that learn from each previous epoch and teaches the model on what went well and what went wrong, and that is done by altering the weight and bias of each node. For our optimizer we use Adam Optimization algorithm, and it does exactly what was said above, it updates the network weights iteratively (through each epoch) based in training data.

After all that we are set and done, we specify how many epochs we want (too low means we will have a very high loss, but too high will get overtrained and each epoch won't contribute to better results), and we found that from 20 to 100 epochs was the sweet spot hitting 98% accuracies consistently.

We then test our model, with the test data (20% of the dataset, the other 80% was used for the training), we evaluate the model and the get loss and accuracy using model.evaluate(). This value shouldn't be used to determine the real accuracy of our model when compared to other models, for that we should use an evaluation data, normally its around 10% of the total dataset, but since we're using a synthetic dataset the best thing to do would be to generate a new set of data.

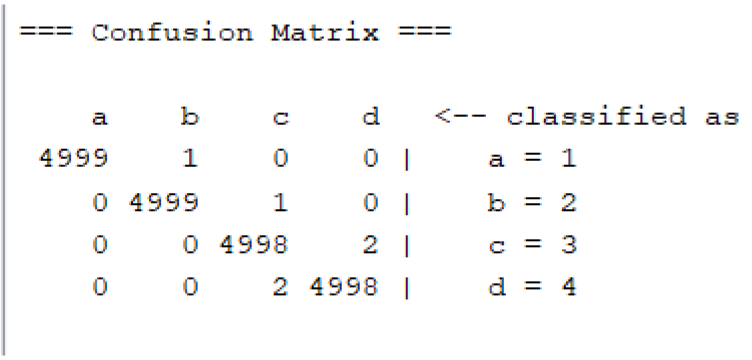
And finally, we save our model as JSON and each nodes weights in binary, we convert our model to be used in the browser using tensorflowJS.converter.save_keras_model(), so it can be loaded and used on the frontend.

Decision tree – J48

J48 is another classification algorithm, it creates a decision tree. The decision tree will have the least possible number of nodes and tend for the best percentage of correct classifications. The chosen test was cross-validation, we choose a specific value for K (number of folds that our dataset will suffer) and the dataset will be split and divided in similar size K times. It will train K times where on the first iteration the first fold will be the testing data and the rest of the folds will be the training data, on the next iteration the second fold will be the testing data and the rest will be the training, and so on until all the folds have been used as testing data.

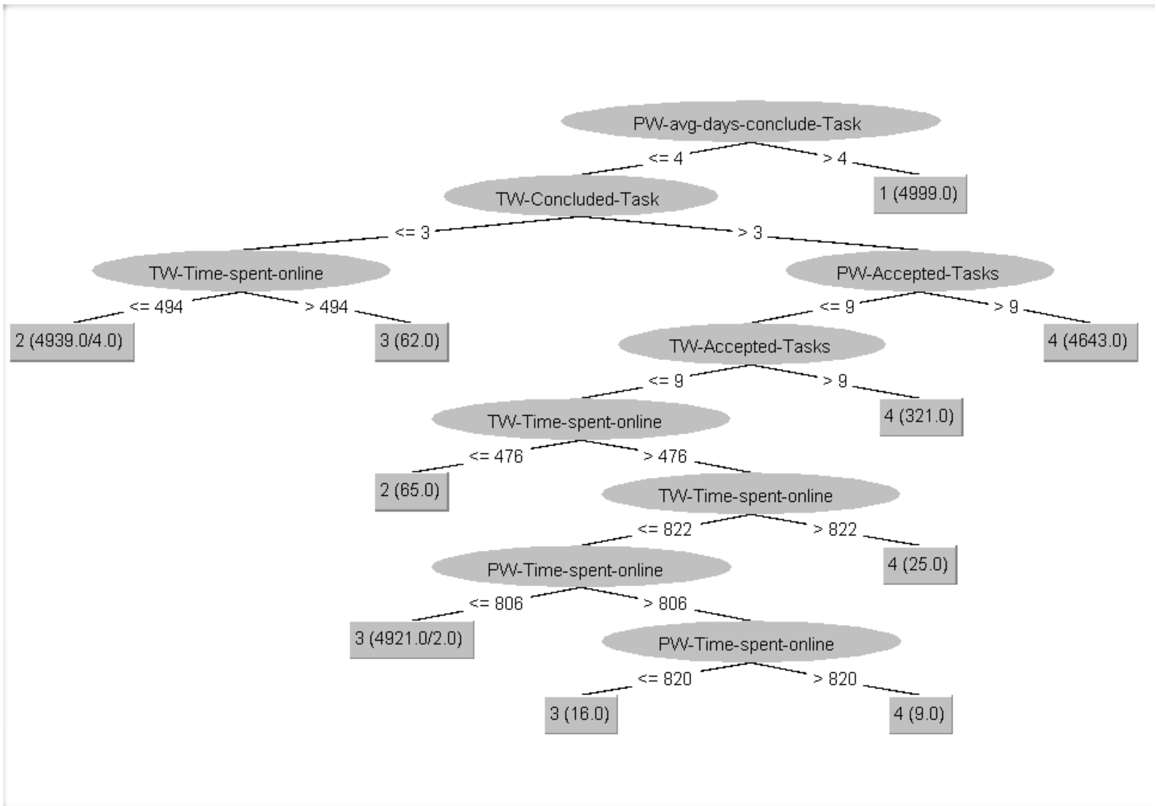
The best K-fold we found was k=11, but honestly any k had a very good accuracy (98% and above), the reason for this is because probably our dataset doesn't overlap enough. The same thing affected the tree being pruned or not, it had no difference on accuracy with either.

On the confusion matrix we can see the results were very good, only failing to correctly predict 6 instances, making it a 99,7% accuracy rate. Below we have a representation of the tree.



```
=== Confusion Matrix ===

    a    b    c    d   <-- classified as
 4999    1    0    0 |    a = 1
    0 4999    1    0 |    b = 2
    0    0 4998    2 |    c = 3
    0    0    2 4998 |    d = 4
```



Output caption: 1-> Churn and busy, 2-> Churn and active, 3-> No Churn and busy, 4-> No churn and active

3. Testing

Test plans and objectives

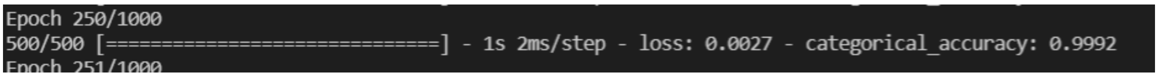
The main objectives to test are the model's performance, to see if inside our classification model we're using the best functions in order to get the best accuracy and the lowest losses.

Model testing

Epochs

For testing purposes, we tested all models with 20 only epochs for a better time optimization.
We have tested with other epochs, but since the increase was minimal from 20 onwards, we decided to just do 20 epochs since its fast and gets great results with our dataset.

Here's an example of epoch 250, only gets a 0.0017 accuracy increase, with a greater risk of overtraining.



```
Epoch 250/1000
500/500 [==============================] - 1s 2ms/step - loss: 0.0027 - categorical_accuracy: 0.9992
Epoch 251/1000
```

Activation functions

We tested three activation functions, ReLU, Sigmoid and softmax.

For a simple neural network, without any hidden layers, with softmax activation function on the output player, these were the test results of changing the activation function on the input layer:

ReLU:

Validation loss: 0.013041824102401733

Validation accuracy: 0.9975000023841858

Softmax:

Validation loss: 1.3902275562286377

Validation accuracy: 0.2512499988079071

Sigmoid

Validation loss: 0.7200815081596375

Validation accuracy: 0.49050000309944153

Now with one hidden layer:

Input layer ReLU and hidden layer ReLU

Test loss: 0.8422929048538208

Test accuracy: 0.4937500059604645

Input layer ReLU and hidden layer Softmax

Test loss: 1.3962990045547485

Test accuracy: 0.25174999237060547

Input Layer ReLU and hidden layer Sigmoid

Test loss: 1.4142944812774658

Test accuracy: 0.2475000023841858

Input Softmax with hidden layer ReLU

Test loss: 0.04344136267900467

Test accuracy: 0.9932500123977661

Input Softmax with hidden layer Sigmoid

Test loss: 1.3868820667266846

Test accuracy: 0.25325000286102295

Input Sigmoid with hidden layer ReLU

Test loss: 0.012546687386929989

Test accuracy: 0.9962499737739563

Input Sigmoid with hidden layer Sigmoid

Test loss: 0.010424967855215073

Test accuracy: 0.9965000152587891

Input Sigmoid with hidden layer Softmax

Test loss: 0.41068848967552185

Test accuracy: 0.9852499961853027

2 hidden layers

For this one, we only tested with ReLU activations

Test loss: 1.387938380241394

Test accuracy: 0.2462500035762787

Learning Rate

The *learning rate* is a hyperparameter that controls how much to change the model in response to the estimated error each time the model weights are updated.

Learning rate 1:

Test loss: 1.3977563381195068

Test accuracy: 0.25224998593330383

Learning rate 0.1:

Test loss: 1.3899046182632446

Test accuracy: 0.2472500056028366

0.01:

Test loss: 0.011914841830730438

Test accuracy: 0.9975000023841858

0.001

Test loss: 0.006891660392284393

Test accuracy: 0.9982500076293945

0.0001

Test loss: 0.39453381299972534

Test accuracy: 0.902999997138977

Optimizers

Optimizers are mathematical functions that depend on the models parameters like weight and biases and they help to know how to change them to reduce the loss calculated.

Adam:

Test loss: 0.013041824102401733

Test accuracy: 0.9975000023841858

SGB:

Test loss: 1.2806516885757446

Test accuracy: 0.34549999237060547

4. Software Construction

Developing this software was very focused on testing, it felt like trial and error, especially the model training part. As seen above, on the model testing chapter, we had to test with epochs, activation functions, learning rate, optimizers and even loss functions (which we didn't include there because categorical cross-entropy was the only relevant one we found), we used tensorflow for the model since it was a requirement set on the beginning, but we used tf.keras, a layer of keras on top of tensorflow so our code would be readable and organized, you can see our version of a keras-less model here:https://github.com/ElynoLamar/IntelligentAwareness/blob/master/code/models/ML/v1/churn.py .We had issues with how the data was going to be shown, since we could use matplotlib to draw charts on python, but our front end was developed in javascript, but we found canvasJS that allows us to replicate some of the matplotlib functionalities allowing us to create charts and show data. Another struggle was using the model we trained to show predictions on the browser, which required extra research on how to convert the model to be saved, loaded and used on the browser. But the biggest one was choosing what dataset we would use, because after it was shown we quickly developed an algorithm for a creation of a synthetic dataset.

5. Software Requirements

Functional requirements

L-> Low, M-> Medium, H->High

#	Requirement name	Description	Pri.
FR01	Homepage	The System must have a home page that introduces the System to users, explains what the Platform does, who it is for, what is the need for it to exist.	L
FR02	Dataset	The system should allow the developer to generate a dataset, by setting up the parameters and their weights	H
FR03	Communities	The system should be able to divide their users into communities	M
FR04	Events	A created event should be attributed to a specific community	M
FR05	Tasks	Tasks are events that should be attributed to a specific target	M
FR06	Predicition	According to the targets performances on completing tasks, the system should make a prediction on their churn and activity rate	H
FR07	Prediction dashboard	The system should have a page where a user can input data and make a prediction according that data	H

FR08	Progress Dashboard	The system should have a page that shows graphs and lets the user filter those graphs by the churn and activity rate	M

Nonfunctional requirements

#	Nome do Requisito	Descrição	Pri.
NFR01	Web development	It should be developed in a way that supports a browser access, with HTML/CSS/JavaScript	H
NFR02	Machine learning enviroment	Should be coded in a programming language that supports this, for an example python	M
NFR03	Database	Should have a database to support data from tasks, communities, targets etc	H
NFR04	Routing server	There should be a middle-ware in between the frontend and database, we will use nodejs	H
NFR05	Personal data	The system shouldn't have to gather personal data, and protect it when its needed.	M
NFR06	Language	For a better generalization, the project should be in english	L

6. Ethics

Analysis

The project itself is built with the best intentions of helping these senior adults to better their lifestyle by supporting them with motivation and socialization. But our intentions never are enough, and we should analyze all the potential negative and positive aspects of releasing this product to the world.

We do not control what the machine predicts, but we control the data we put there to be trained. One of the biggest problems is related to bias, a single value of data could be enough to change from one prediction to another. In today's age, there aren't many systems that run only on Machine learning, they normally require human supervision, but even then, the human that's supervising might be affected by a wrong prediction.

*Algorithms introduced in non-medical fields have already been shown to make problematic decisions that reflect biases inherent in the data used to train them. For example, programs designed to aid judges in sentencing by predicting an offender's risk of recidivism have shown an unnerving propensity for racial discrimination (*Conway, M., & O'Connor, D. (2016)) *[7].*

Another big problem to tackle is the governance problem, for machine learning to work there must be a dataset and normally a dataset is either gathered from forms, interviews, and datamining, but the biggest issue with that is the transparency of the data gathering, many people aren't aware or warned about what will be done with their information and data, which of course isn't ethical. There is not many laws and entities to enforce that what these companies are doing with the data is not being abused.

Consequentialism

On the Ethic of consequentialism, we value the consequences of the actions. It depends on the accuracy of the predictions and how it affects the platform user's perceptions, if by getting a prediction and the help needed to better their life actually works then it would be ethical, on the other hand, if it had repercussions and the prediction would bring the person even more down, then it wouldn't.

Deontological Ethic

On the Deontological Ethic, we value the intentions of the actions, and our project has the best intention in mind, reducing this unwanted loneliness so they can have better mental, physical and emotional health.

Threats

One of the biggest threats would be if we or anyone else used our algorithm to track and find those who are more vulnerable. Thankfully our model and our dataset only work specifically for this issue, so other people can't use this model to find these vulnerable people through methods like social media datamining and such.

7. When Data Science Becomes Software Engineering

In this project we had to take the role of both a software engineer and a data scientist. Firstly, let's see what the difference is between them

Software Engineer: is a person who applies the principles of *software engineering* to the design, development, maintenance, testing, and evaluation of computer *software*.

Data science: work on data collected to build predictive models and develop machine learning capabilities to analyze the data captured by the software.

Make data scientists implement and more especially learn methods used in software engineering, below we have a list of SE goals from SWEBOK:

1. Correctness
2. Reliability
3. Modifiability
4. Testability
5. Reusability
6. Maintainability
7. Efficiency
8. Usability
9. Portability
10. Interoperability

We tried to tackle some of these goals, specifically the efficiency and testability goals.

8. Conclusion

In conclusion, this project was very interesting because it tackled a real situation, and it could change the world a little for the better. Without model training experience and starting, we had to learn a lot and we're satisfied with the results. If more time was given, more algorithms would be tested, and a better dataset would be generated in order to have more variety and less bias.

9. References

[1]Ahmad, A. K., Jafar, A., & Aljoumaa, K. (2019, March 20). Customer churn prediction in telecom using machine learning in big data platform. Retrieved from https://journalofbigdata.springeropen.com/articles/10.1186/s40537-019-0191-6#:~:text=Three machine learning algorithms were,Networks to predict churn factor.&text=The AUC values were 99.10,and support vector machine, respectively.

[2]Brandusoiu I, Toderean G, Ha B. Methods for churn prediction in the prepaid mobile telecommunications industry. In: International conference on communications. 2016. p. 97–100.

[3]Angwin J, Larson J, Mattu S, Kirchner L. Machine bias. ProPublica. 2016 May 23; (https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing)

[4]Morag McDermont, Helen Thomas, Marilyn Howard (2018): Isolation and Loneliness of Older People. https://doi.org/10.5523/bris.xmql2x6m5mws24mw3y8im7mwy

[5]Ayastuy, M. D., Torres, D., & Fernández, A. (2020, December 09). Adaptive gamification in Collaborative systems, a systematic mapping study. Retrieved from https://sciencedirect.com/science/article/pii/S1574013720304330

[6]MacDonald KJ, Willemsen G, Boomsma DI, Schermer JA. Predicting Loneliness from Where and What People Do. Social Sciences. 2020; 9(4):51. https://doi.org/10.3390/socsci9040051

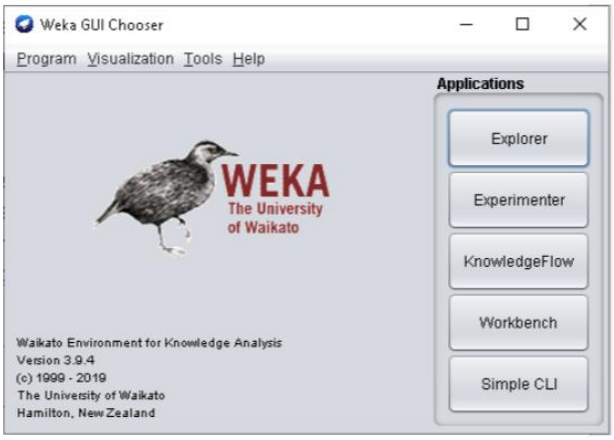
[7] Conway, M., & O'Connor, D. (2016). Social media, big data, and mental health: Current advances and ethical implications. *Current Opinion in Psychology, 9,* 77–82. https://doi.org/10.1016/j.copsyc.2016.01.004

10. Attachments

Weka walkthrough

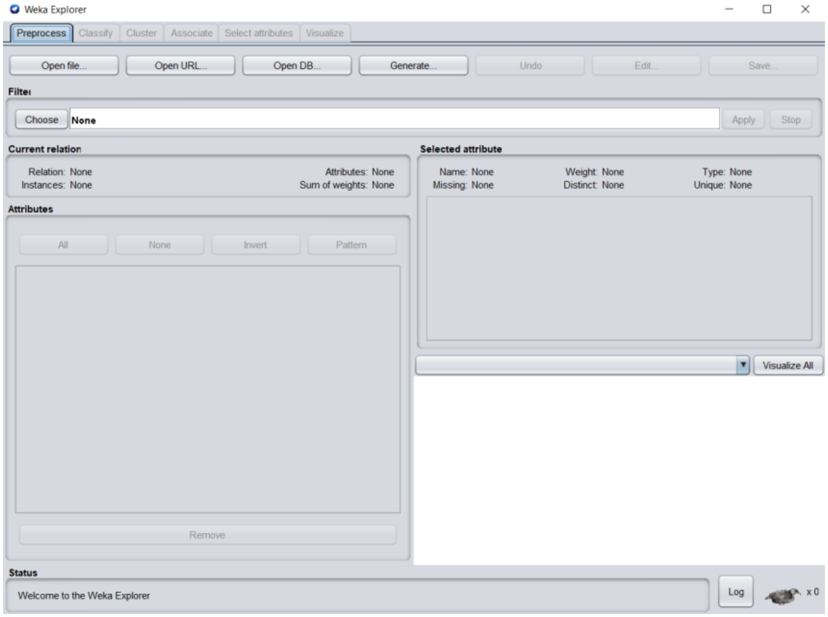
First, download weka as an executable from : https://waikato.github.io/weka-wiki/downloading_weka/

After it is installed, you will be faced with this window:



Waikato Environment for Knowledge Analysis
Version 3.9.4
(c) 1999 - 2019
The University of Waikato
Hamilton, New Zealand

Then, choose explorer and the next window will pop up:



Filter: None

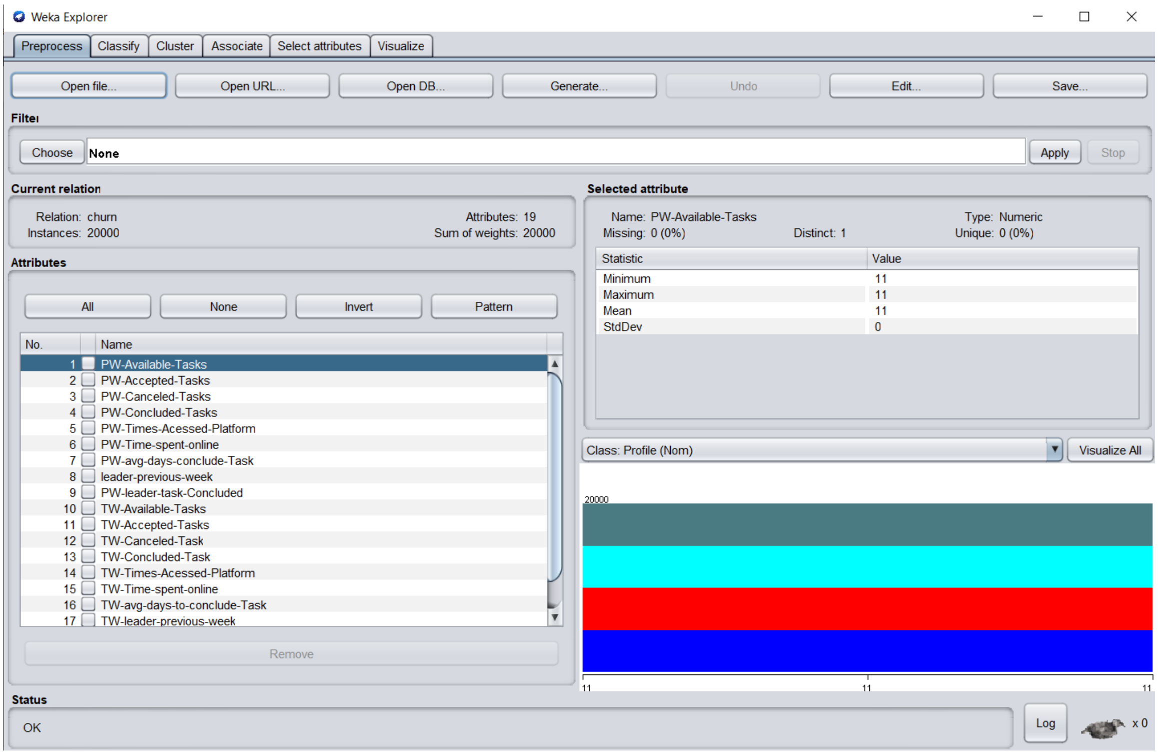
Relation: None
Instances: None
Attributes: None
Sum of weights: None

Name: None
Missing: None
Weight: None
Distinct: None
Type: None
Unique: None

Welcome to the Weka Explorer

Now on the open file you can upload your dataset, you may use the same one we used for this project by accessing out github (https://github.com/ElynoLamar/IntelligentAwareness/blob/master/code/models/ML/dataset.data)

After it is loaded, you will see this window:



Filter: None

Current relation

Relation: churn
Instances: 20000
Attributes: 19
Sum of weights: 20000

Selected attribute

Name: PW-Available-Tasks
Missing: 0 (0%)
Distinct: 1
Type: Numeric
Unique: 0 (0%)

Statistic	Value
Minimum	11
Maximum	11
Mean	11
StdDev	0

Attributes

No.	Name
1	PW-Available-Tasks
2	PW-Accepted-Tasks
3	PW-Canceled-Tasks
4	PW-Concluded-Tasks
5	PW-Times-Acessed-Platform
6	PW-Time-spent-online
7	PW-avg-days-conclude-Task
8	leader-previous-week
9	PW-leader-task-Concluded
10	TW-Available-Tasks
11	TW-Accepted-Tasks
12	TW-Canceled-Task
13	TW-Concluded-Task
14	TW-Times-Acessed-Platform
15	TW-Time-spent-online
16	TW-avg-days-to-conclude-Task
17	TW-leader-previous-week

Class: Profile (Nom)

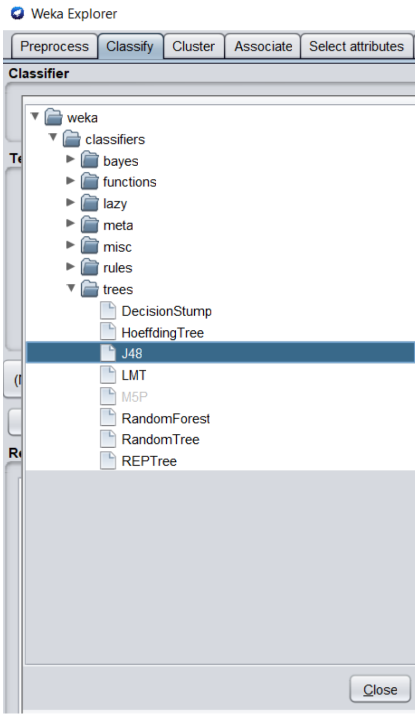
Status

OK

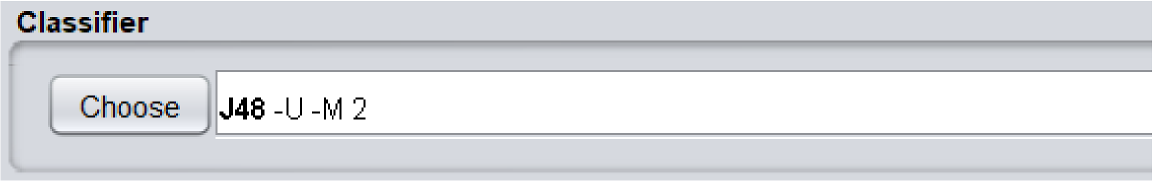
By pressing the different attributes you will filter the graph to show you each KPI by profile amount(1(blue)-> churn and busy, 2(red)-> churn and active, 3(cyan)-> no churn and busy, 4(grey)->no churn and active).

Then, press the classify tab on the top.

Press the classify button and choose the trees branch and choose the j48 algorithm.



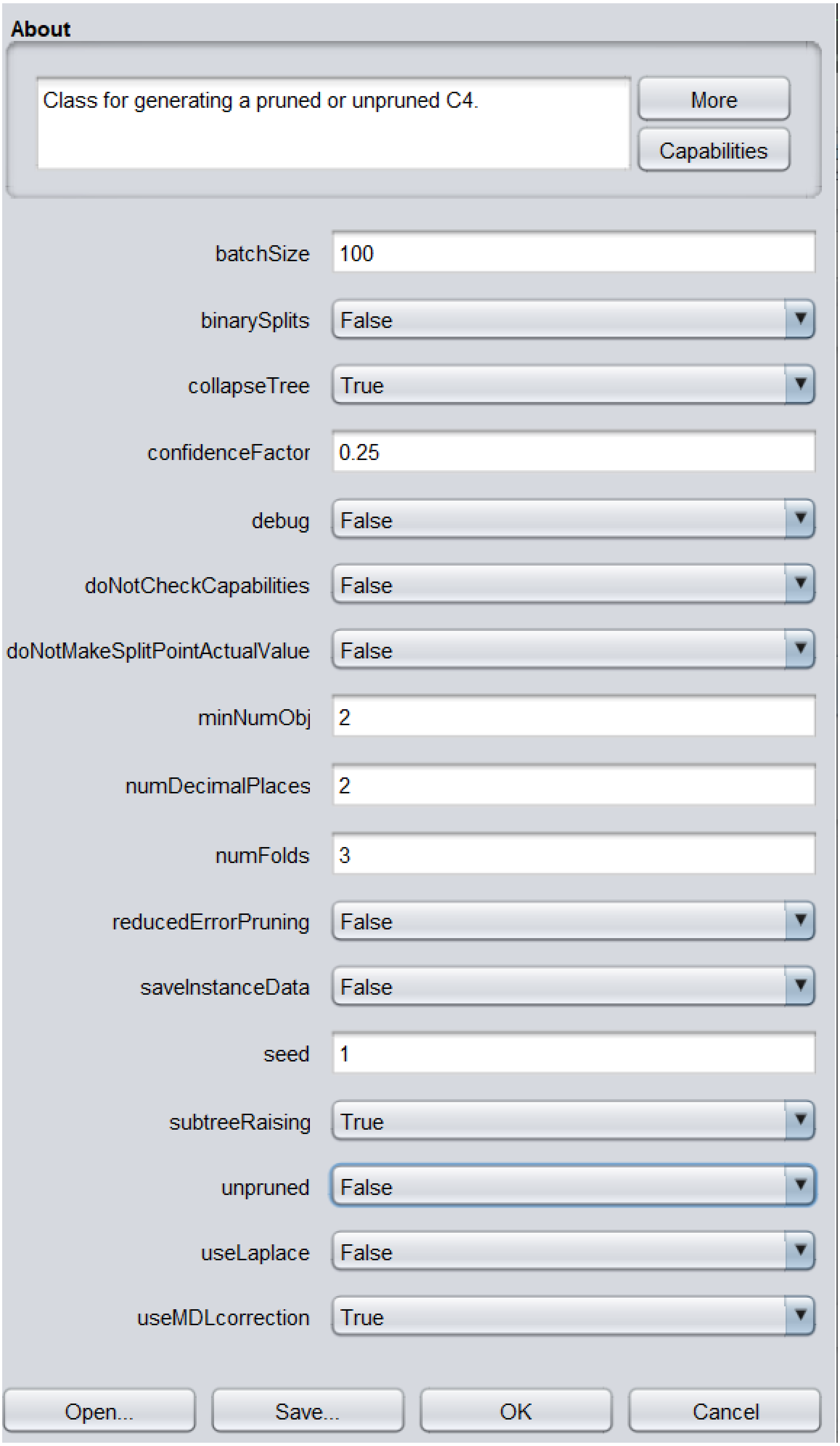
Then we can click on the algorithm name to define some parameters, by default weka pre-defines them.



Classifier

J48 -U -M 2

This next window will pop-up, the most important parameter is the unpruned(true/false), by setting it to false, means we will be using pruning and the decision tree will ignore irrelevant branches, by setting it up to true, we will get the full tree.

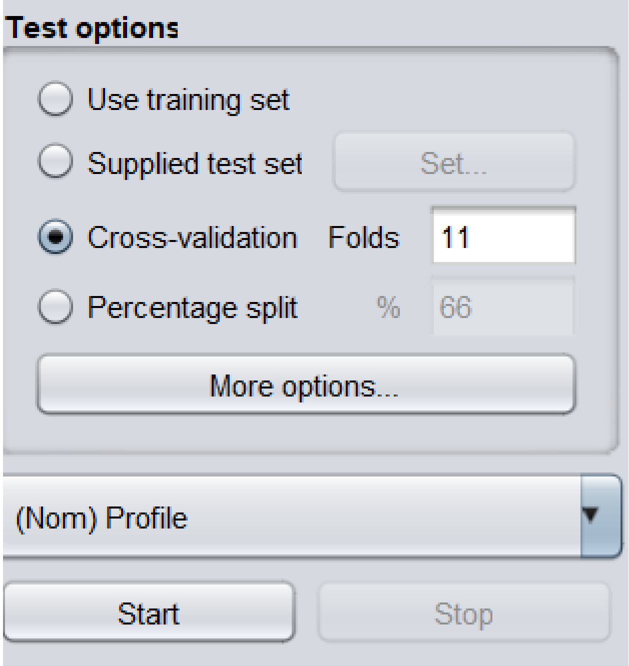


About

Class for generating a pruned or unpruned C4.

batchSize	100
binarySplits	False
collapseTree	True
confidenceFactor	0.25
debug	False
doNotCheckCapabilities	False
doNotMakeSplitPointActualValue	False
minNumObj	2
numDecimalPlaces	2
numFolds	3
reducedErrorPruning	False
saveInstanceData	False
seed	1
subtreeRaising	True
unpruned	False
useLaplace	False
useMDLcorrection	True

Then you choose how many k-folds the dataset will have, set it up to 11 and then press start



Test options

○ Use training set

○ Supplied test set

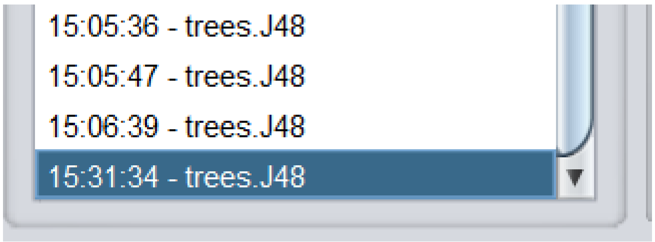
● Cross-validation Folds 11

○ Percentage split % 66

(Nom) Profile

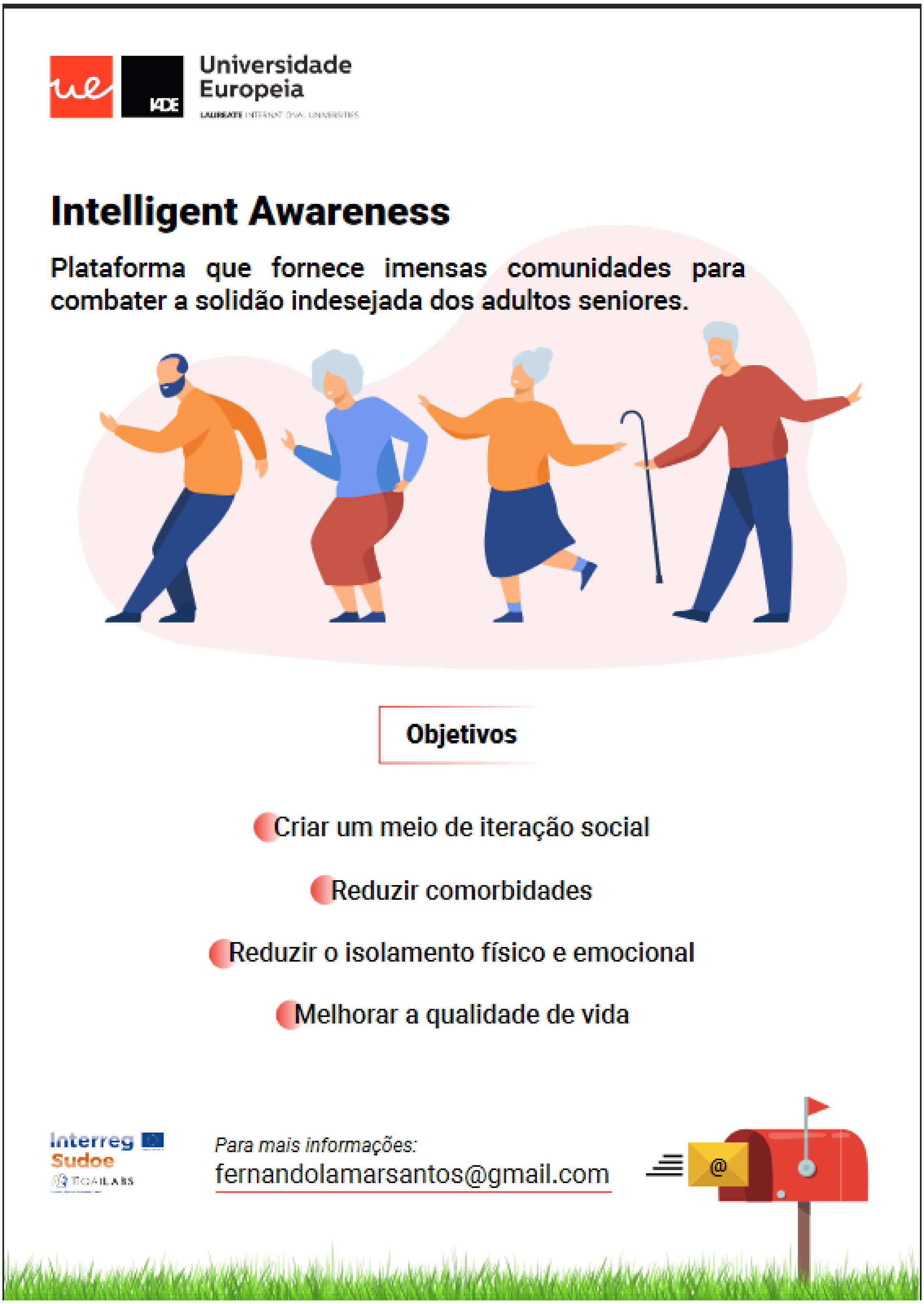
And done, on the main window you will have the classification output, it will show your dataset columns and amounts, a text version of your tree, and on the summary it will show the accuracy of the model and a confusion matrix so you can easily see how many instances were incorrectly or correctly classified.

If you want a better visualization of the decision tree, right click the algorithm on the result list and press the visualize tree option.



```
15:05:36 - trees.J48
15:05:47 - trees.J48
15:06:39 - trees.J48
15:31:34 - trees.J48
```

Posters



Universidade Europeia

Intelligent Awareness

Plataforma que fornece imensas comunidades para combater a solidão indesejada dos adultos seniores.

Objetivos

- Criar um meio de iteração social
- Reduzir comorbidades
- Reduzir o isolamento físico e emocional
- Melhorar a qualidade de vida

Para mais informações:
fernandolamarsantos@gmail.com

Interreg Sudoe

MOAILABS

European Regional Development Fund

Investigação e inovação

“Criar redes de apoio para pessoas que vivem sozinhas”

MOAI Labs

Laboratórios de Inteligência Coletiva e Tecnologia Social e de Saúde para combater o isolamento e a solidão de pessoas idosas

Mais informação em
www.moailabs.eu

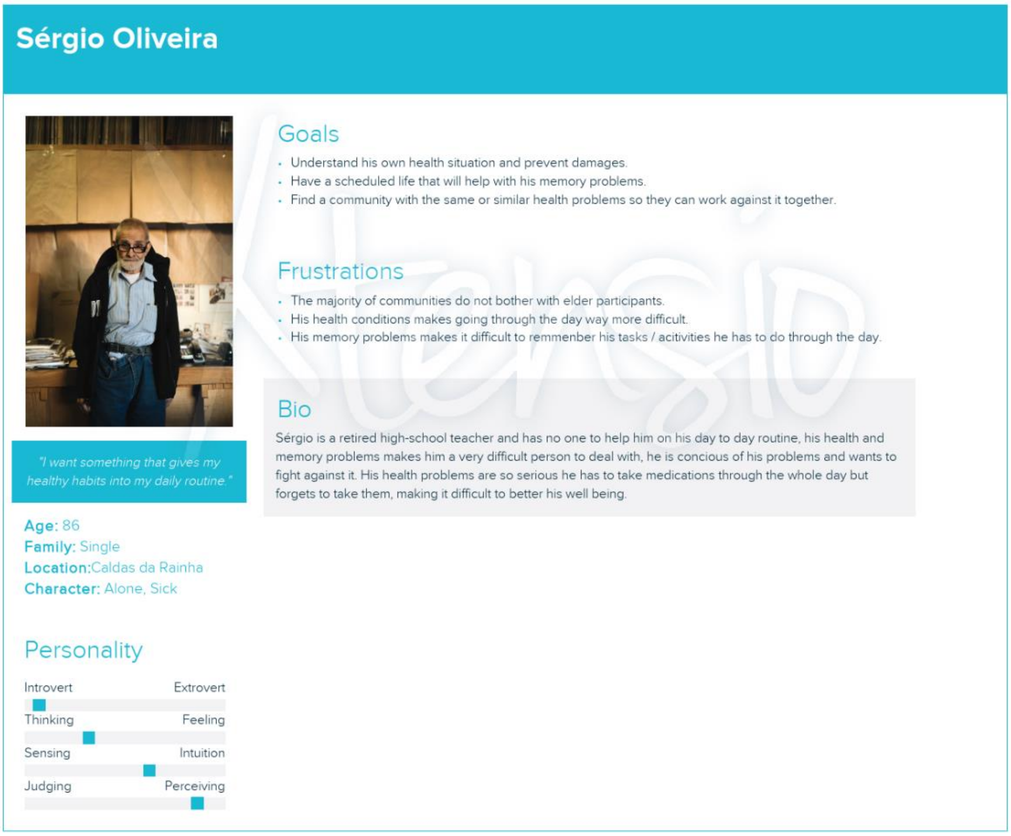
Fundación intras · Junta de Castilla y León · Servicios Sociales · FUNGE UVa · LEITAT · i2mi · universidade de aveiro · CINTESIS Health Research · INOV · ticbiomed

www.interreg-sudoe.eu

Projeto financiado pelo Programa Interreg Sudoe
através do Fundo Europeu de Desenvolvimento Regional (FEDER)

www.moailabs.eu

Personas for profile creation



Sérgio Oliveira

Goals

- Understand his own health situation and prevent damages.
- Have a scheduled life that will help with his memory problems.
- Find a community with the same or similar health problems so they can work against it together.

Frustrations

- The majority of communities do not bother with elder participants.
- His health conditions makes going through the day way more difficult.
- His memory problems makes it difficult to remmenber his tasks / acitivities he has to do through the day.

Bio

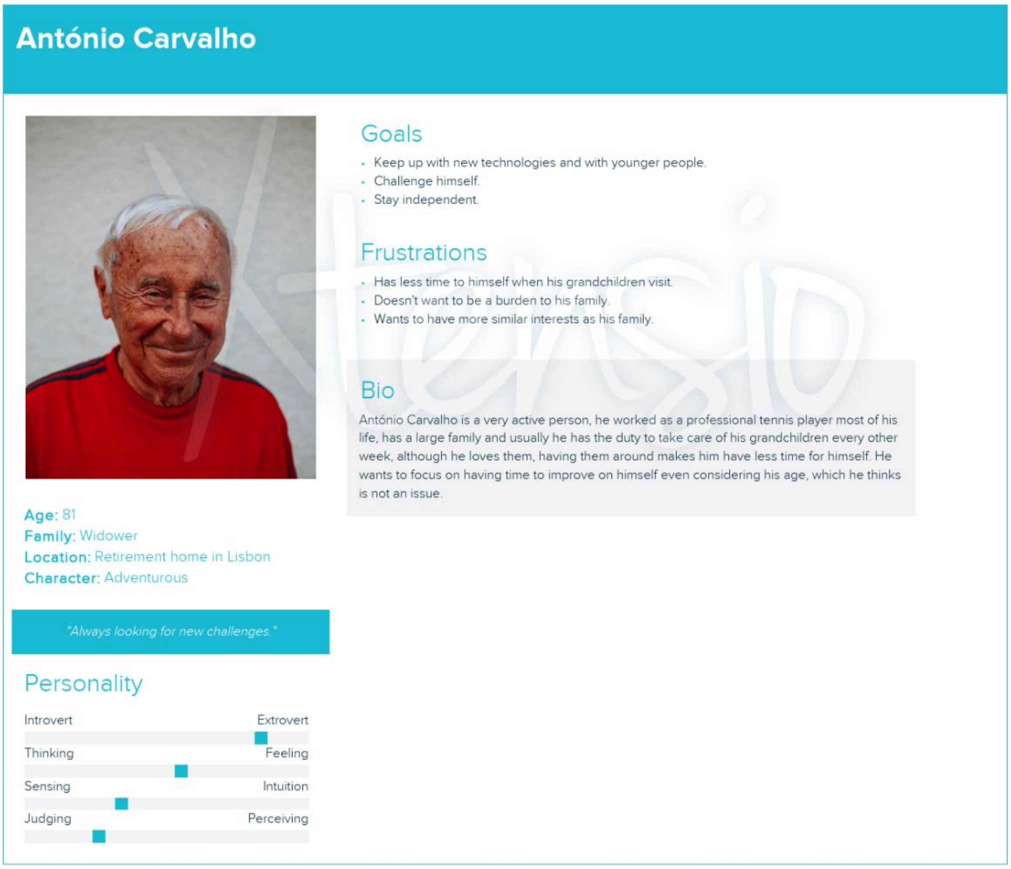
Sérgio is a retired high-school teacher and has no one to help him on his day to day routine, his health and memory problems makes him a very difficult person to deal with, he is concious of his problems and wants to fight against it. His health problems are so serious he has to take medications through the whole day but forgets to take them, making it difficult to better his well being.

"I want something that gives my healthy habits into my daily routine."

Age: 86
Family: Single
Location: Caldas da Rainha
Character: Alone, Sick

Personality

Introvert — Extrovert
Thinking — Feeling
Sensing — Intuition
Judging — Perceiving



António Carvalho

Goals

- Keep up with new technologies and with younger people.
- Challenge himself.
- Stay independent.

Frustrations

- Has less time to himself when his grandchildren visit.
- Doesn't want to be a burden to his family.
- Wants to have more similar interests as his family.

Bio

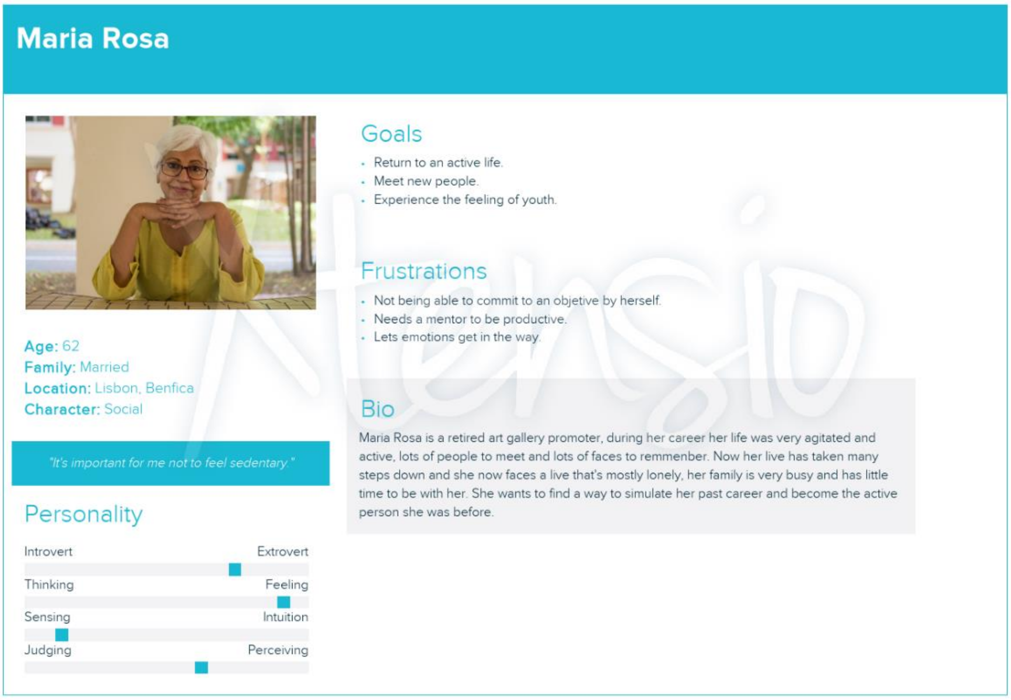
António Carvalho is a very active person, he worked as a professional tennis player most of his life, has a large family and usually he has the duty to take care of his grandchildren every other week, although he loves them, having them around makes him have less time for himself. He wants to focus on having time to improve on himself even considering his age, which he thinks is not an issue.

Age: 81
Family: Widower
Location: Retirement home in Lisbon
Character: Adventurous

"Always looking for new challenges."

Personality

Introvert — Extrovert
Thinking — Feeling
Sensing — Intuition
Judging — Perceiving



Maria Rosa

Age: 62
Family: Married
Location: Lisbon, Benfica
Character: Social

"It's important for me not to feel sedentary."

Personality

Introvert	Extrovert
Thinking	Feeling
Sensing	Intuition
Judging	Perceiving

Goals

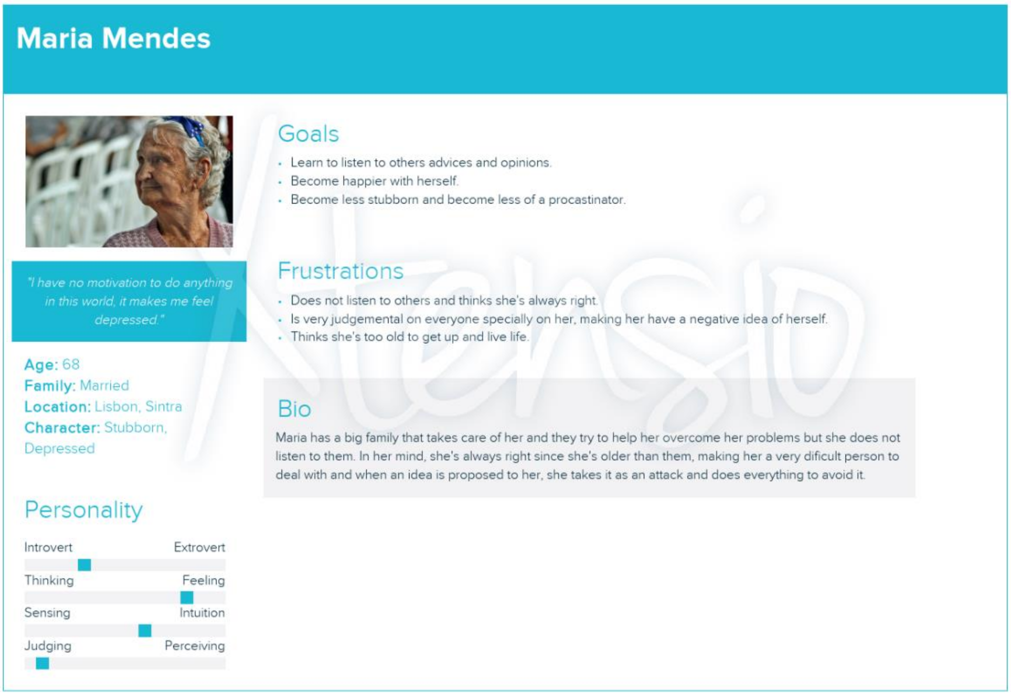
- Return to an active life.
- Meet new people.
- Experience the feeling of youth.

Frustrations

- Not being able to commit to an objetive by herself.
- Needs a mentor to be productive.
- Lets emotions get in the way.

Bio

Maria Rosa is a retired art gallery promoter, during her career her life was very agitated and active, lots of people to meet and lots of faces to remmenber. Now her live has taken many steps down and she now faces a live that's mostly lonely, her family is very busy and has little time to be with her. She wants to find a way to simulate her past career and become the active person she was before.



Maria Mendes

"I have no motivation to do anything in this world, it makes me feel depressed."

Age: 68
Family: Married
Location: Lisbon, Sintra
Character: Stubborn, Depressed

Personality

Introvert	Extrovert
Thinking	Feeling
Sensing	Intuition
Judging	Perceiving

Goals

- Learn to listen to others advices and opinions.
- Become happier with herself.
- Become less stubborn and become less of a procastinator.

Frustrations

- Does not listen to others and thinks she's always right.
- Is very judgemental on everyone specially on her, making her have a negative idea of herself.
- Thinks she's too old to get up and live life.

Bio

Maria has a big family that takes care of her and they try to help her overcome her problems but she does not listen to them. In her mind, she's always right since she's older than them, making her a very dificult person to deal with and when an idea is proposed to her, she takes it as an attack and does everything to avoid it.