A Trend in Global Happiness
Bobbie van Gorp & Beau Furnée
Machine Learning (AUC): Final Report
22-12-17

Introduction

Since 2012, the World Happiness Report has compared the happiness levels for every country in the world. It is used by governments to measure progress and adapt policies. For our research, we analyzed these happiness scores in order to for example improve the effectiveness of implementing new policies.
For our research, we used the dataset from 2015, since it was complete and additional data for this year was easily accessible via the internet.

Approach

The first step we had to take was to complete our dataset, which was also a very time consuming step. Besides converting the data in the original dataset to numbers that we could use in our analysis, we also wanted to add more features. Our final dataset consisted of the following features:

- Country ID
- Continent ID
- Happiness Score
- GDP per Capita
- Life Expectancy
- Unemployment Rate
- Literacy

We used multiple online datasets for this, which can be found under 'Sources'.
The next step was to choose the algorithms we were going to apply to the dataset. To begin with, we chose to use linear regression, since it is a fairly elementary and generally-applied algorithm, so it would make a strong starting point.
Since we expected our dataset to be nonlinear, we thought working with a support vector machine will result in better regression and also provide a good transition into classification.
The happiness scores in the dataset are continuous in the range from 1 to 10. Apart from the regression algorithms that were applied, we decided to also look at this dataset as a classification problem, since this might result in critical new insights to better understand our dataset. This meant that there would have to be a discrete number of groups of happiness scores as labels. For this, the range of happiness scores was divided into 10 clusters. We chose to apply the decision tree algorithm as the classifier for this adapted dataset. During the application of the algorithms, we tried to improve the results through parameter optimization.

	A	B	C	D	E	F	G	H
1	**Country**	**Region**	**Happiness Score**	**GDP per capita**	**Life Expectancy**	**Unemployment Rate (%)**	**Literacy**	**Member of OECD**
2	Afghanistan	3,00	3,575	584,025902	63,29820	35	31,7411	0
3	Albania	9,00	4,959	3954,02278	78,20315	10,3	97,247	0
4	Algeria	6,00	5,605	4132,76029	75,85529	10,5	75,1361	0
5	Angola	2,00	4,033	3695,79375	61,18934	6,7	66,0301	0
6	Argentina	7,00	6,574	13467,1024	76,29302	8,5	98,09	0
7	Armenia	9,00	4,35	3609,65478	74,20620	18	99,7444	0
8	Australia	10,00	7,284	56554,0388	82,45122	5,8	99,0013	1
9	Austria	1,00	7,2	43665,0095	81,84390	6	98,1003	1
10	Azerbaijan	9,00	5,212	5500,31038	71,84520	5,9	99,7881	0
11	Bahrain	6,00	5,96	22688,8782	76,86520	4,1	94,5568	0
12	Bangladesh	3,00	4,694	1210,15812	72,22161	4,1	47,0773	0
13	Belarus	9,00	5,813	5949,11068	73,62439	0,8	72,7587	0
14	Belgium	1,00	6,937	40356,875	81,28780	7,9	99,3021	1

To better understand our results, we visualized them in graphs and generated scores. This way, we were not only able to assess each algorithm individually, but also compare them to each other.
In the next part of this report we will evaluate our analysis, and justify the decision we made concerning matters like algorithm, data, and parameter choices.

Methods
Before applying the algorithms, we normalized the dataset and divided it into a training and testing set.

Linear Regression
The linear regression algorithm from the 'scikit learn' library doesn't have any parameters or variables to optimize, so we simply used the standard algorithm.

Support Vector Machine
In order to implement this machine learning algorithm, the SVR (Epsilon-Support Vector Regression) class from the scikit learn module was used. Using the GridSearchCV class, different values of the C (regularization parameter), epsilon (epsilon-tube within there is no penalty for points predicted within a distance epsilon from the actual value) and tol (the tolerance for the stopping condition) parameters were tested on cross validation sets using 4 fold cross validation in order to optimize these parameters. Apart from this, different kernel types were tested, namely the linear, polynomial, radial basis function and sigmoid kernels, and for the polynomial kernel, different degrees were tested. It turned out, however, that the 'standard' radial basis function kernel was the optimal kernel as it returned the highest cross validation R^2 score. The optimized parameter values were 0.8, 0.07 and 0.001 for the C, epsilon and tol parameters respectively.

Decision Tree
As was mentioned before, the happiness score range was divided into 10 clusters to be able to use this algorithm. This was done by rounding off the happiness scores of the countries so that they could be used as labels for the testing and training data. The decision tree algorithm that was used was the DecisionTreeClassifier from scikit learn and this class has different parameters that could be optimized. First it was tested whether the best criterion, or the function that measures the quality of a split, was the standard gini impurity criterion or the entropy (information gain) criterion. The gini impurity criterion turned out to return the highest mean accuracy on the 10 fold cross validation sets. The values for the other parameters max_depth (maximum depth of the decision tree) and min_samples_leaf (minimum number of samples required to be at a leaf node) were also optimized using the GridSearchCV class and their optimal values turned out to be 5 and 15 respectively.

Graphing
Since we wanted to visualize the performance for all the individual features, we had to take some additional steps. To get correct line in the graph, the data needed to be ordered ranging from lowest to highest, otherwise the line would move between points seemingly randomly. First, we tried converting our dataset from a numpy array to a structured array. Although this allowed us to sort the array based on every feature, it proved to be difficult to transform the structured array back to an array that we could apply our algorithms on. Therefore, we decided to order the data in the original Excel file and

create 5 different datasets. Because these datasets were only to be used for graphing, we trimmed them down to keep the graphs readable.
Per feature, we graphed the feature value and the according happiness score as red crosses. Next, we added the values the algorithms predicted per feature as a line, to show whether the algorithm was able to recreate the general trend to the data.

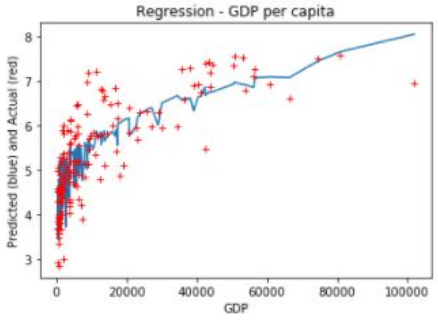
Results

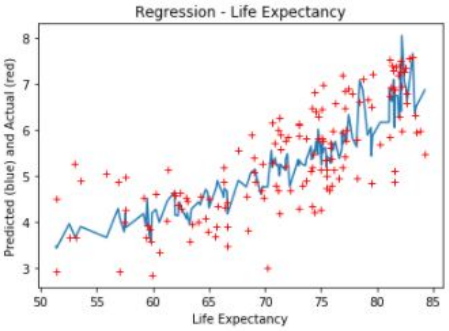
In order to clearly see the results of the algorithms, the predicted and actual happiness scores were plotted against the four main features (GDP per capita, life expectancy, literacy and unemployment rate) and these graphs are shown below. Especially in the GDP per capita and the life expectancy features we can see that the predictions of the algorithm show a clear trend, but for the other features this is more difficult to see. This might also be the reason that even our best classifier, the SVM with an R^2 score of 0.66142 on the test set, does not show an impressive results: from the features that were chosen, there is not always a clear trend in the data.
The other regression algorithm, linear regression, has a relatively similar performance with an R^2 score of 0.6039 on the test set, with a similarly clear trend for the GDP per capita and life expectancy predictions.
The decision tree classifier, however, shows very bad performance in the 'artificial' classification problem, with a mean accuracy score of only 0.42. Of course this is still much better than random guessing, which would result in an accuracy of around 0,1, but it is definitely not an impressive or valuable performance. The confusion matrix of the decision tree classifier is shown below.

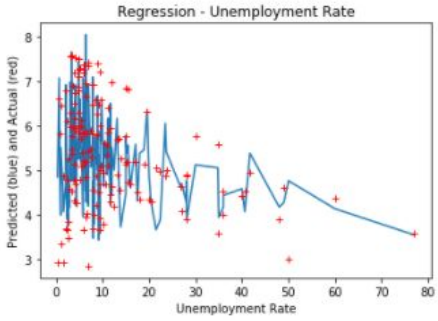
```
[[0 1 0 0 0 0]
 [0 2 1 0 0 0]
 [0 1 6 3 0 0]
 [0 1 3 4 0 0]
 [0 0 1 6 0 3]
 [0 0 0 0 0 3]]
```

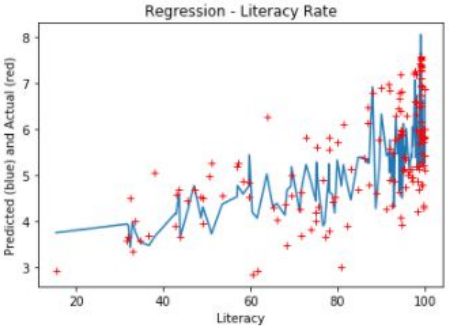
This confusion matrix only shows the values for the classes 2 to 7 since there were no predictions or actual scores in any of the other classes. In the matrix we can see that the classifier is hardly ever completely off, but its predictions are often one class apart from the actual values.

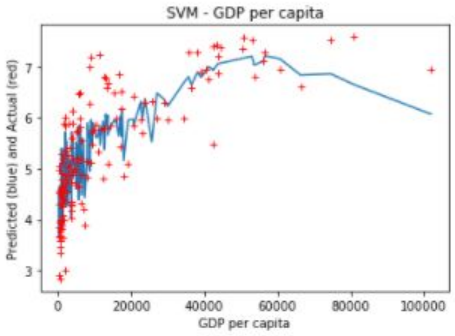
Graphs

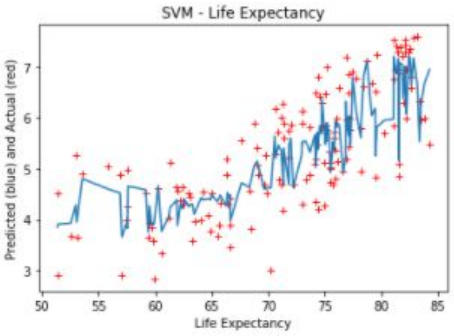

Regression - GDP per capita
Predicted (blue) and Actual (red)
GDP

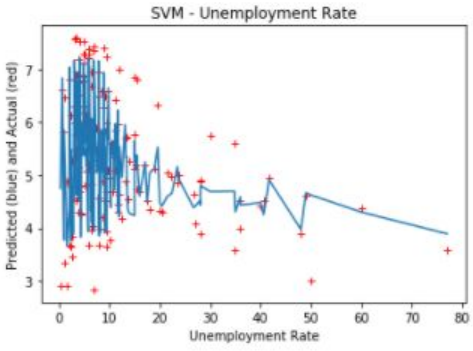

Regression - Life Expectancy
Predicted (blue) and Actual (red)
Life Expectancy

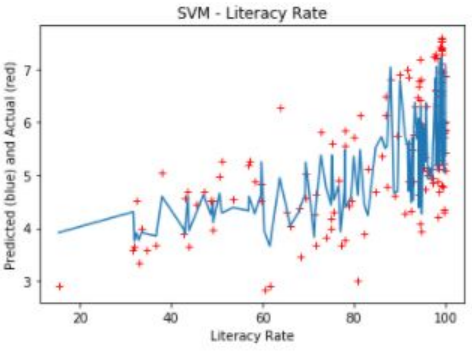

Regression - Unemployment Rate
Predicted (blue) and Actual (red)
Unemployment Rate


Regression - Literacy Rate
Predicted (blue) and Actual (red)
Literacy


SVM - GDP per capita
Predicted (blue) and Actual (red)
GDP per capita


SVM - Life Expectancy
Predicted (blue) and Actual (red)
Life Expectancy


SVM - Unemployment Rate
Predicted (blue) and Actual (red)
Unemployment Rate


SVM - Literacy Rate
Predicted (blue) and Actual (red)
Literacy Rate

Evaluation

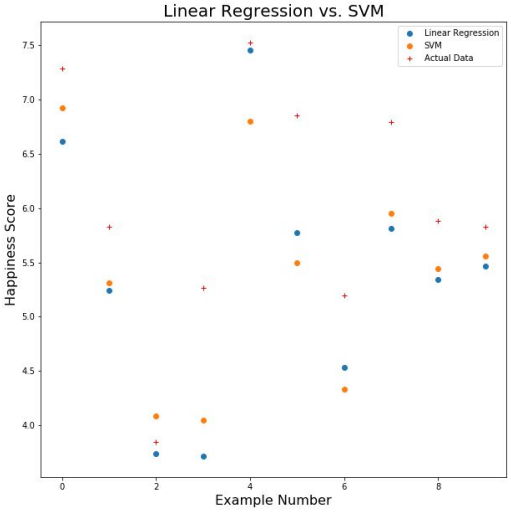
When overlaying the graphs from the two regression algorithms, we end up with the graphs shown below.

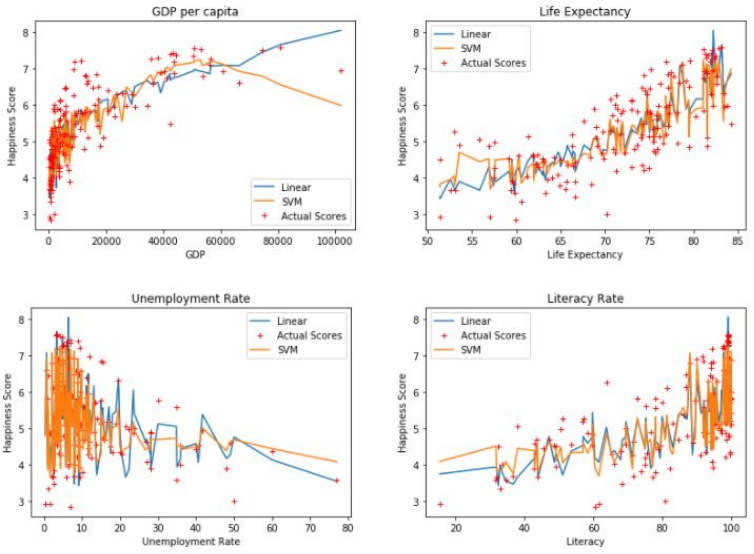
When looking at the scores, it proves that there is a lot of room for improvement. Both the regression algorithms came up with a similar score, so it is difficult to pick one over the other between linear regression and SVM. Especially when comparing the predicted scores with the actual scores, we can conclude that the predicted scores are closer to one another than to the actual value.

However, it is clear from the score of the decision tree that it is not beneficial to use that algorithm. This does not directly imply that decision tree is a bad algorithm, but it does mean that it is for our dataset. We assumed that we could create clusters for our continuous dataset, but it clearly did not work out. Decreasing the amount of clusters would probably increase the accuracy, as the predictions were often very close to - but not the same as - the actual class.

The results of this research give us valuable information and insights about our approach. In machine learning, we are interested in data predictions by algorithms, because these algorithms can process many more features and show relations between features that are impossible to see for humans, because there is simply too much information to process. Simple data predictions and relations can be made and seen by people, such as the relation between GDP per capita and happiness shown in the graphs of this research. This research shows that when there are both not many (training) examples and not many features available, the added value of machine learning algorithms over human predictions is not very significant. The added value becomes much larger when there is much more data available.

In order to make accurate world happiness predictions, it is essential that many more features are selected and obtained so that the machine learning algorithms can do what they do best: extract relations from large complicated datasets and make predictions for data when there is simply too much information for humans to process and oversee.





Sources

https://data.worldbank.org/indicator/NY.GDP.PCAP.CD

https://data.worldbank.org/indicator/SP.DYN.LE00.IN

https://www.kaggle.com/unsdsn/world-happiness/data

https://www.cia.gov/library/publications/the-world-factbook/rankorder/2129rank.html

https://knoema.com/atlas/

https://www.ncbi.nlm.nih.gov/pubmed/27437333

https://tradingeconomics.com/

https://data.worldbank.org/indicator/SE.ADT.LITR.ZS