Comparing Machine Learning Algorithms that Predict Age from a Person's Humour Style

[redacted]

Date: 26/01/2022

[redacted]

Abstract— As a result of a range of social impacts, our use of humour can vary significantly. These impacts will change amongst different generations, hence we can model the relationship between a population's age and the type of humour they tend to use. In this paper, I have used data collected from a "development of humour styles" questionnaire, which categorises an individual's humour into four types, to create three predictive models. I have decided to treat the participants' age as both continuous and discrete data, in order to assess and compare a wide range of supervised machine learning algorithms. Although classification and regression algorithms calculate age through different metrics, I found that both find difficulty in predicting age, however it is clear that polynomial regression is able to produce the best model - for reasons to be discussed.

Index Terms— Classification – modelling where a discrete category/label is predicted after learning from a training set of input-output pairs, hyperparameter – a specific configuration of a parameter that is set and tuned for a learning algorithm, quantile binning – setting boundaries for data which are not fixed but include an equal/similar portions of data called bins, regression – modelling where a specific value from a continuous distribution is predicted after learning from a training set of input-output pairs, supervised learning – a process by which a function/mapping of input-output pairs is inferred through many pieces of labelled data (a training set); examples include classification and regression models

1 Introduction

IN this study, I will be exploring three machine learning algorithms which all aim to answer the question: "can we predict an individual's age from the combination of humour they tend to use?"

I was able to create these models with the use of data collected through a Humour Styles questionnaire [1]. This questionnaire took a random sample of members of the population and asked each member several questions, in order to score them on how much affiliative, self-enhancing, aggressive and self-defeating humour they tend to use. Additionally, their age, gender and a subjective answering accuracy were all recorded.

With the use of this data, I decided to define the four humour scores as a multivariate independent variable / input, in addition to "age" as the dependent variable / output for my three predictive models - since that is what we are attempting to predict.

In answering the same question and using the same data, I can fairly evaluate and compare each of my models to one another. This question holds significance since we can discover what type of relationship exists between a person's humour style and their age. The effect of age in humour psychology is currently up for contention, as evidence points at both younger individuals using more aggressive and self-defeating humour and there being no differences between ages at all [2]. Out of the models I have chosen, we can learn which one best predicts the relationship out of the data I have selected, however, first I must detail my data preparation process.

2 Exploratory Data Analysis and Data Preparation

2.1 Data Cleaning and Data Construction

The key characteristics of the data can be seen through the density distributions below.

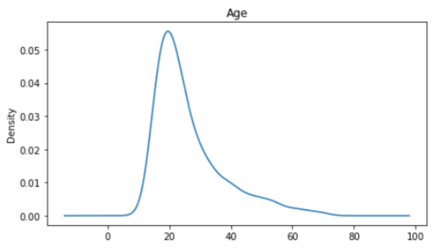


Fig 1. The majority of the participants from our sample are just above 20 years of age. There is a positively skewed normal distribution.

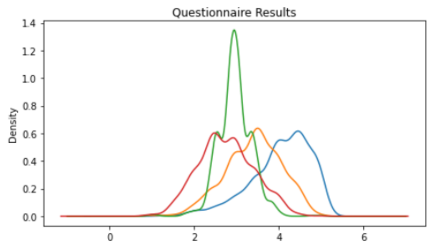


Fig 2. The distributions of the four humour types are between scores of zero and six. Each follow a rough normal distribution.

As we can see, each of the data categories we are interested in fit under a normal distribution curve, which is useful information when deciding which method to transforming our data with.

To ensure that our data is as accurate as possible, I decided to fix in place the bounds of "age", the questionnaire results and "gender". In addition, I removed any rows with an accuracy of zero since these people did not want to be included in research. I also assumed that all the calculated humour results from the questionnaire were correct and that they followed the formula set out in Martin et al.'s study.

Since we want to make sure that the data we are training our models with is as accurate as possible, I decided to remove any data rows with an accuracy below 80, since this would leave 75% of our original data we could use - as seen below.

```
count    1063.000000
mean       87.499530
std        12.044556
min         2.000000
25%        80.000000
50%        90.000000
75%        95.000000
max       100.000000
```

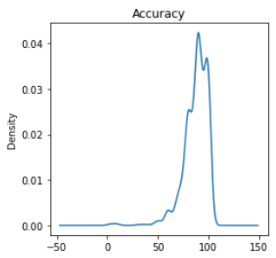


Fig 3. 25% of data lies between the range of an accuracy of 2 and 80.

I could have added a weighting to each data-pair when training my models however these values are subjective and so our model would still not be completely accurate. Usually, data with a percentage error of ~10% is good practice, however the figure I chose to remove had to be a compromise between having accurate data and having enough data - both allowing us to train more accurate models.

In addition to removing these rows, the original dataset also included unnecessary columns (including gender, the questions and now accuracy), which I also removed since we are focusing on the humour results and the participants' age. At this point, our dataset looks like the following.

	affiliative	selfenhancing	aggressive	selfdefeating	age
0	4.0	3.5	3.0	2.3	25
1	3.3	3.5	3.3	2.4	44
3	3.6	4.0	2.9	3.3	30
4	4.1	4.1	2.9	2.0	52
7	4.4	4.1	3.3	2.5	34

Fig 4. Dataset after row and column reduction, containing continuous ages

Having now reduced the size of the dataset, I split the data into the features X (the humour values) and our target Y (age) – each stored in the form of an array. I also shuffled the data to allow models to be created through random arrangements of input-output pairs, however I did this by using a constant random seed, allowing us to fairly compare the models to each other later.

2.2 Data Construction Continued and Data Transformation

Having defined our X and Y variables, we can see the data we currently have follows the characteristics below.

	affiliative	selfenhancing	aggressive	selfdefeating	age
count	912.000000	912.000000	912.000000	912.000000	912.000000
mean	4.047478	3.399013	2.964254	2.757237	26.487939
std	0.705502	0.660247	0.391589	0.652356	11.257288
min	1.300000	1.000000	1.000000	0.900000	14.000000
25%	3.600000	3.000000	2.800000	2.300000	18.750000
50%	4.100000	3.500000	3.000000	2.800000	23.000000
75%	4.600000	3.900000	3.100000	3.100000	31.000000
max	5.100000	5.000000	5.000000	5.000000	70.000000

Fig 5. A statistical summary of the data currently stored as our X and Y variables, which has a lower total count than before.

Since our variables follow a bell-shape curve, as determined previously, we can scale/transform the data to have a mean and standard deviation of zero. This ensures our variables are on the same scale for the learning algorithms and will help us to achieve better model performance [3].

After performing this process to both our shuffled X and Y variables, I split this data into a training set, a validation set, a combined training and validation set, and a testing set – having dedicated 70% of the data to our training set to provide enough in the training process. I decided to create my own validation set, even though this is created with hyperparameter testing later, in order to test an initial/dummy model. Additionally, similarly to the questionnaire, I assumed that the sets I created were representative of the data as a whole.

Instead of treating "age" as a continuous variable, we could also treat it as a discrete one by creating a set of age ranges. Treating our output variable in this way opens us up to a wider selection of learning algorithms. To create these discrete sets, I decided to use quantile binning since our data is skewed to one side; this ensures that we create a number of bins containing the same frequency of data. I created 7 bins, since this nicely separated our age bounds into distinct integers; in this instance our bins were defined at ages 14, 17, 19, 22, 24, 29, 39 and 70. I then transformed these age brackets into numbers, leading to the data table below.

	affiliative	selfenhancing	aggressive	selfdefeating	age_binned
0	4.0	3.5	3.0	2.3	4
1	3.3	3.5	3.3	2.4	6
3	3.6	4.0	2.9	3.3	5
4	4.1	4.1	2.9	2.0	6
7	4.4	4.1	3.3	2.5	5

Fig 6. Dataset after quantile binning of age, leading to values referring to each data bin.

Like previously, I also scaled our new binned age values to have a mean and standard deviation of zero, which this time was done by shifting each value down by three (since we have seven bins containing equal frequencies). Finally, I split these into a new training set, validation set, combined training and validation set, and testing set, for this alternate output we are also interested in finding.

3 Learning algorithms Implementation

3.1 Learning Algorithm Selection

I decided to select three supervised learning algorithms, since I had planned to supply labelled training data [3], to generate a model which maps the humour score inputs to an output of age. Since there are multiple features / inputs, we must provide our models with multivariate data.

I am interested in treating "age" as both a continuous variable – and so will be implementing two regression algorithms – as well as a discrete variable – hence I will also implement one classification algorithm. All these algorithms will help me create sufficient models to answer my question.

The first of these models is linear regression, which provides a linear mapping from X to Y. It uses a cost function, to measure how the model performs on the entirety of the training set, and uses the concept of gradient decent, allowing us to adjust the hypothesis function which represents the model. Vectorisation is used so that our algorithm can operate on a set of values at one time, since we have multivariate data. The second model is polynomial regression, which is rather similar, however uses the concept of higher order relationships. This can help us avoid the issue of creating a model which underfits our data, but instead we may see overfitting. Both of overfitting and underfitting will lead to poor predictions of data. Our final model is the k-nearest neighbours (KNN) algorithm, which creates a classification model. This uses the idea of Voronoi cells, which classify our data to a particular set, depending on a certain distance measure. Now that I have discussed the algorithms I will implement, we can move onto the model training.

3.2 Model Training and Evaluation

When training each of my models, I followed a similar process. This included defining a dummy model, as a test to see what would happen when instantly predicting the validation data after being fit to the training data. Then I defined another instance of the model, in order to perform hyperparameter tuning with cross validation. Here I defined the hyperparameters, after finding the model's parameters I was interested in, and the process of cross validation – detailing the split of validation data and training data (the k-fold) and the number of repetitions in different arrangements. I fitted the hyperparameter tuning model to the training data to find the optimal hyperparameters. Finally, I defined an instance of the best model, which included these optimal hyperparameters, and trained it on the validation and training data, and set it to predict the testing data [3].

To train the linear regression model, I simply created an initial model and fitted this to the validation and training data. I then used this model to predict the testing data. In this case, I didn't apply hyperparameter tuning, since the parameters didn't directly affect the model. They included calculating the model's intercept, normalising, copying the X variables and the number of jobs used for computation. None of these parameters would add to the predictive optimisation of this model. Since linear regression only attempts to seek a linearly relationship between data, this model may encounter underfitting, however in this circumstance this was not seen.

To train the polynomial regression model, as discussed previously, I created a dummy model. The only parameter we are interested in is the polynomial degree, since this is the only change that will directly affect the model; the other parameters are shared with our previous linear regression model. As a result, I set this as our hyperparameter to be tuned. The candidate values I set were between 2 and 4, to help us avoid overfitting, since higher polynomials lead to overfitted data, and so that our algorithm would have a faster time efficiency. To perform this, I created a pipeline to link our polynomial features and our model. I then defined GridSearchCV (which performs the hyperparameter tuning) using cross-validation with 4 folds and setting the performance metric to the negative mean absolute error (MAE). GridSearchCV checks through every combination of hyperparameters and so would be less likely to settle upon a higher polynomial to overfit the data, which a random search may have done. I decided upon using the negative MAE, since the negative root mean squared error (RMSE) gives high weights to large errors, whereas the negative MAE is able to measure a closer accuracy [3]. I then trained this on polynomial transformed X training and Y training values. This process resulted in a polynomial degree of 2 to be found, which hence provides the best polynomial mapping between X and Y. Finally I defined another model, taking a degree of 2; I trained this on the polynomial transformed training and validation data and then attempted to predict the testing data.

To evaluate both these regression models, I calculated the MAE and mean squared error (MSE). These are appropriate since they provide the cost function for regression. I decided upon analysing both since MAE is more robust with outliers, but it still doesn't work too well with imbalanced data [3].

To train the KNN algorithm, I once again created a dummy model and set the parameters I was interesting in to random values. These parameters included the number of neighbours (classifications), the leaf size (the maximum number of points belonging to each leaf node), the p score (the power parameter for the distance metric) and the distance metric. I decided to use a uniform weight function, so that all the points in a neighbourhood were weighted equally. I also let the specific algorithm type be automatically decided. To avoid overfitting and underfitting, we must ensure that our k-value is just right - by using cross-validation once again [3] we can ensure that the newly tested data won't encounter this problem, due to its repeated testing. Consequently, I defined another model and then set the candidate values of the hyperparameters to include: a leaf size between 1 and 50, a neighbour size between 1 and 30, a p score of either 1 or 2 and a distance metric of either Minkowski, Hamming, Manhattan or Euclidean (which are all the metrics used for KNN). I defined the GridSearchCV – having selected this once again to try all combinations – implementing the cross validation with four k-folds to avoid any fitting issues (as discussed previously). Instead of setting the performance metric to MAE, since this is for regression evaluations, I used "accuracy" which simply calculates and then provides a classification accuracy score (through dividing the correct model predictions by all the predictions). With this second model, I trained it on the training data, which included the separate binned Y training data. This provided me with the best leaf size of 15, a p score of 1 and 29 neighbours. Finally, I created another model, with these parameter values; I supplied the validation and training data and attempted to predict the testing classification data.

I evaluated this classification model through finding the accuracy (F_1) score and an accompanying classification report. This score may be misleading when using unbalanced data [3], thus I also decided to evaluate this model through creating a confusion matrix, which tells us exactly how many true positives, false positive, false negatives and true negatives we have – allowing us to see exactly how many correct predictions there are and of which type.

4 Model Comparison

The calculated MAE and MSE for the linear regression were 0.855 and 1.27 respectively, to 3 sf. For the polynomial regression, 0.841 and 1.26 were respectively calculated. Below we can visualise what our regression models looks like and how they reached their scores.

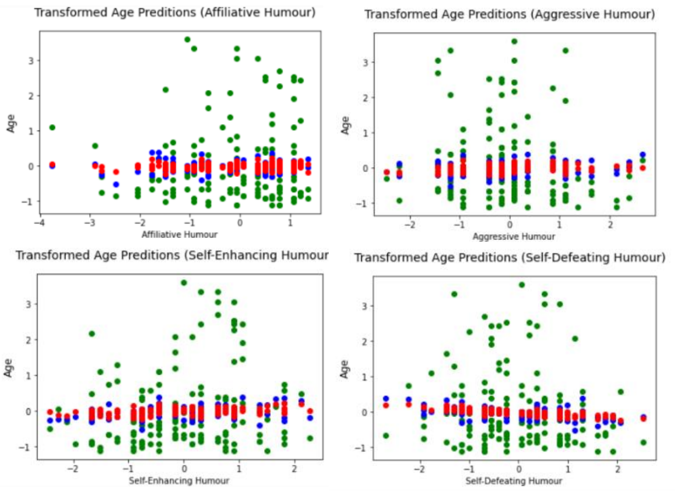
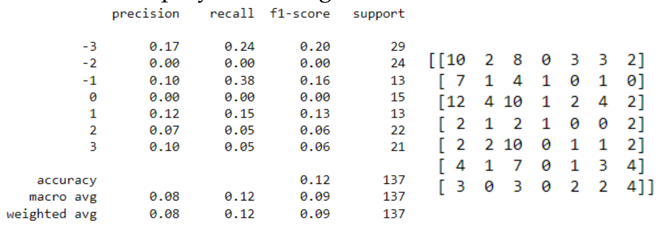


Fig 7. Green = the original testing set, red = linear regression prediction, blue = polynomial regression prediction. All other humour scores have been made constant in each graph.

With these cost scores, we can clearly see that both the models perform in an extremely similar way. However, since the polynomial regression model has a greater vertical spread - and so can encompass more ages - and as its MAE and MSE scores are slightly lower, I would argue that polynomial regression can provide a better model to predicting an age from a person's humour style.

Although a "good" MAE and MSE are specific to our datasets, when it comes to defining an accurate classification model, we simply want to get a high accuracy / F_1 score. As seen in Fig 9a, our model has produced a very low accuracy, with either a complete failure or near-zero failure for every category. When it comes to comparing classification and regression problems, it is rather subjective, since the data we transform and the way use build and evaluate our model differ quite significantly. Nevertheless, as seen with our confusion matrix also, after receiving many false positives and false negatives on the test data, we can be certain that our KNN model is very weak, and so our preference should be one of our regression models – polynomial regression.



```
              precision    recall  f1-score   support

          -3       0.17      0.24      0.20        29
          -2       0.00      0.00      0.00        24
          -1       0.10      0.38      0.16        13
           0       0.00      0.00      0.00        15
           1       0.12      0.15      0.13        13
           2       0.07      0.05      0.06        22
           3       0.10      0.05      0.06        21

    accuracy                           0.12       137
   macro avg       0.08      0.12      0.09       137
weighted avg       0.08      0.12      0.09       137
```

```
[[10  2  8  0  3  3  2]
 [ 7  1  4  1  0  1  0]
 [12  4 10  1  2  4  2]
 [ 2  1  2  1  0  0  2]
 [ 2  2 10  0  1  1  2]
 [ 4  1  7  0  1  3  4]
 [ 3  0  3  0  2  2  4]]
```

Fig 8. KNN classification report, where -3 to 3 are the bins we previously defined.

5 Conclusions and Discussion

In this study, I planned to create and compare three predictive models, through machine learning, to predict a person's age from the type of humour they tend to use.

This process included data cleaning, construction, transformation, model training through cross validation and hyperparameter testing, and evaluating our models in a specific way to the algorithm we used.

I would argue that out of the models I created, polynomial regression proved to be the best at predicting our original question, since its cost value prevails over linear regression and since we found clear issues our KNN model faced.

If I were to repeat this procedure, I would source data that wasn't as skewed, since this resulted in oddly defined data bins for my KNN model. I would also possibly use fewer inputs, or perform dimensionality reduction if I had more time, as these provided trouble when attempting data visualisation and this would have sped up the computation time of my models.

Finally, to provide more concrete analysis, it may have been better to compare models that treated our dependent variable as either a continuous or a discrete variable instead of both, and perhaps it may have been better to choose a simpler question. However, the analysis and question I chose were both interesting, so I am content with the results I have found.

References

[1] Martin, R. A., Puhlik-Doris, P., Larsen, G., Gray, J., & Weir, K. (2003). Individual differences in uses of humor and their relation to psychological well-being: Development of the Humor Styles Questionnaire. *Journal of Research in Personality*

[2] Jiang, F., Lu S., Jiang, T. & Jia, H. (2020). Does the Relation Between Humor Styles and Subjective Well-Being Vary Across Culture and Age? A Meta-Analysis. *Frontiers in Psychology*. <frontiersin.org/articles/10.3389/fpsyg.2020.02213/full>

[3] Shi, L. (2021). AI Lecture PowerPoints 1 – 20. *AI Lectures - COMP2261*, Durham University, delivered 4 October 2021