2019

Group Project

COMP9417 - AARON, BORIS AND ANDREW - LOOKING FOR ONE MORE MEMBER :)

AARON BLACKWELL, ANDREW LAU, COLIN YANG, TSUN MING LEUNG

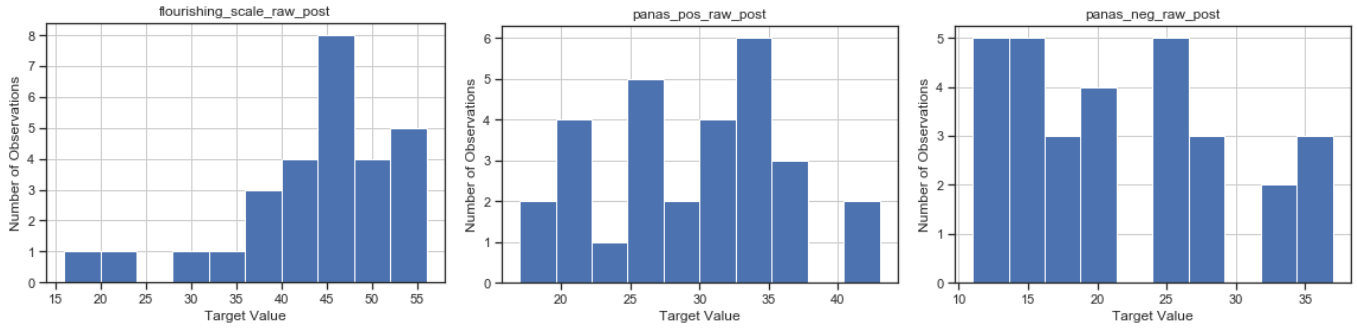
1 Introduction

Our report investigates whether machine learning methods can be applied to student mobile phone sensor data in order to predict post-term measures of psychological wellbeing and identify what features, if any, are import in predicting psychological wellbeing. The StudentLife dataset consists of data gathered as part of a study (Wang, et al., 2014) from the phones of 48 students attending Dartmouth College during a 10-week term as well as pre and post measures of psychological wellbeing (the flourishing scale and PANAS positive and negative scores). Time series data from the sensors were available for each of the 48 students (with varying levels of data completeness) from which features were engineered for modelling. A model search was conducted (fitting an assortment of 10 different machine learning methods), before we settled on investigating regularised generalized linear models (GLM), support vector machines (SVM), k-nearest neighbours (KNN) and gradient boosted ensembles of trees (GBM). A variety of pre-processors (e.g. k-nearest neighbours' imputation and principal component analysis) were also experimented with. Throughout our iterative machine learning cycle, we used the cross-validated training score to select and validate methods alongside diagnostics tools, such as predicted vs. observed charts, Lorenz curves and SHAP feature importance measures before finally evaluating model performance on the test data. Both classification and regression were explored in our analysis.

2 Dataset

The StudentLife dataset consists of 10 types of mobile phone sensor data collected via a smart phone app. There are measures of physical activity/movement, physical state (stationary, walking, running, unknown), audio (silence, voice, noise) and light as well as location, Bluetooth, Wi-Fi and phone charging/locking data in (typically) 10 to 30-minute increments available as a time series for each student. To summarise the data into meaningful features, we grouped our data into weekly chunks and from which features were extracted. The timestamps in the dataset were adjusted to the US/Eastern time zone as indicated by (Wang, et al., 2014) with Monday to be considered the start of each week. We inferred the starting date for the semester (week 1) and implemented it as 2013-03-18.

Figure 1 - Distribution of target values



The above histograms show the distribution of the target values. We can see that the flourishing scale is left skewed, the PANAS positive score vaguely Gaussian and the PANAS negative score possibly right skew. We have converted the raw scores into classification targets by separating that scores using the median. The use of the median to divide the scores will give us balanced classes, avoiding a potential problem in classification of the model being biased towards the majority class (Blagus & Lusa, 2013).

2.1 Missingness

As identified by the StudentLife study authors, there were issues with student compliance and data quality, particularly for the sensor data towards the end of the term.

2.1.1 Sensor data

In the Figure 2 below, we can see that the non-compliance rate amongst students increases to around 20% of all the students as the term progresses. These periods of inactivity/no sensor input decrease the signal to noise ratio of our features, negatively impacting model performance. These periods of inactivity also distort the time-based features (e.g. time spent stationary) that use the time elapsed between adjacent observations, creating unrealistically large values of time elapsed. These erroneous values were controlled for by capping the time elapsed at the 98th percentile. Figure 3 below highlights the levels of missingness for our features. The missing data also creates issues when using summations and counts as weekly summary statistics as they would be capturing exposure/compliance rather than the construct the sensor is trying to capture. Where appropriate, we have strived to use ratios and measures of central tendency (mean, median) to avoid conflating exposure with the construct the feature is trying to measure.

Figure 2 - Periods of missing activity

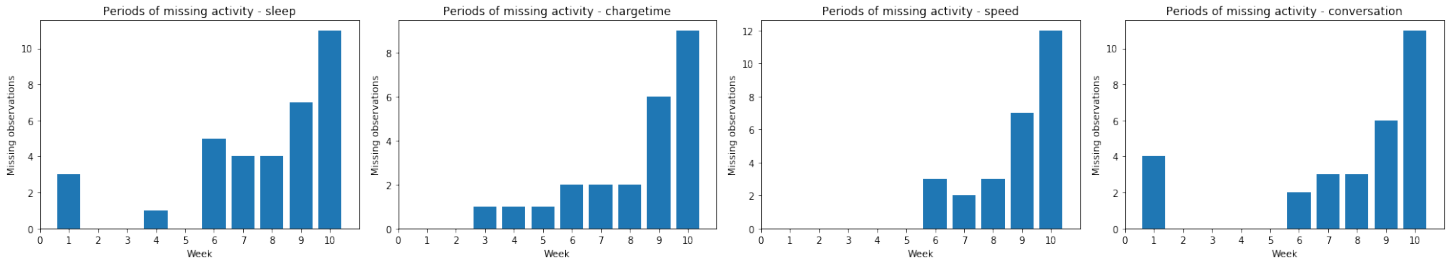
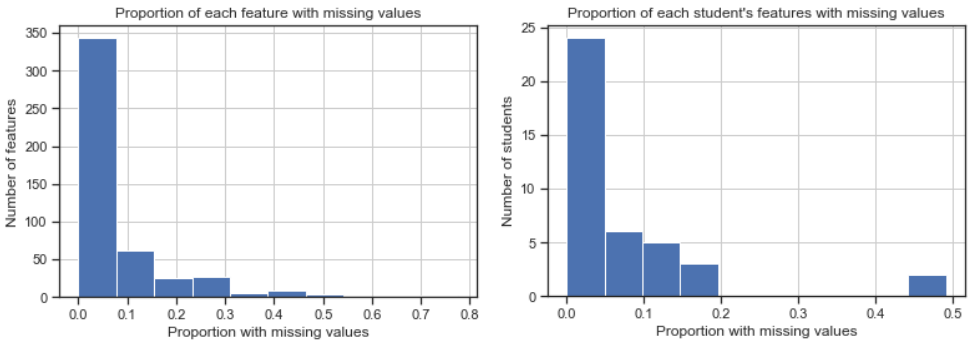


Figure 3 - Proportion of missing values



2.1.2 Target data

The target data (flourishing and PANAS scores) also suffered from missing data issues. The flourishing scale, PANAS positive score and PANAS negative score had 6, 4 and 6 missing records respectively. Given our dataset only consists of 48 records of which on 30 participants completed post observation surveys, this represents a significant degree of missingness in our target data.

2.1.3 Correlation Heatmaps

Correlation heatmaps were created in order to identify potentially important features as well potentially confounding variables and interactions to explore. See Appendix page 16.

2.2 Feature Extraction and Engineering

Sleep feature – Sleep has an important impact on mental health. We based our sleep feature on the previous work of identifying sleep patterns in (Chen, et al., 2013) who use a linear regression model to estimate total sleep. To provide reasonable results we provided some additional checks for sleep, such that at least 3 of the dark sensory, phone-lock, phone-charge, stationery and silence features must be active at a given time period and the period of sleep must last at least 15 minutes.

The sleep feature was aggregated to a daily level and then summarised to the min, max, mean and median sleep hours per day each week. The sleep feature has three main limitations in being used; first it is only an approximation, second, the aggregations do not make adjustments for different sleeping patterns and the impact of changing days during a sleep period, and thirdly there is no adjustment for periods where students do not have their phone power on and on their person.

Phone lock and phone charge features – In the raw data, periods of time where the phone is locked or being charged for more than 1 hour was recorded. We produce a number of features for the min, max, mean and median time a phone is locked and/or charged each day of the week. It is expected that phone lock and phone charge features can act as proxies for sleep time.

Wi-fi and Location features – Measures of movement (speed and distance), type of movement, altitude, time spent at different locations, time spent indoors and outdoors as well as compass bearing were extracted from the wi-fi and location time series. These give us measures of the levels of indoor and outdoor activity which may have an important impact on mental health. Appropriate summary statistics were produced at the weekly level for each of these features (e.g. counts, min, max, mean, median and standard deviation).

Bluetooth feature – The number of distinct Bluetooth devices detected around a person can indicate sociability and exposure to other individuals which has been found to be correlated with flourishing and PANAS positive scores (Chen, et al., 2013) . Counts of the number of distinct Bluetooth devices daily were aggregated at the weekly level to produce the daily average number of distinct Bluetooth devices per week.

Activity feature – In the raw activity data, there are three classes – stationary, running and others. Running can be seen as an indicator of sports activity and exercise. Exercise has been shown to improve mental health by reducing anxiety, depression and negative mood (Sharma, Madaan, & Petty, 2006). The proportion of time the user is stationary and running for each week were extracted as features.

Conversation feature – Conversation frequency and total conversation duration can be used to indicate social interaction and level of sociability. Conversation frequency and total conversation duration for each week were extracted as features.

3 Methods

3.1 Summary

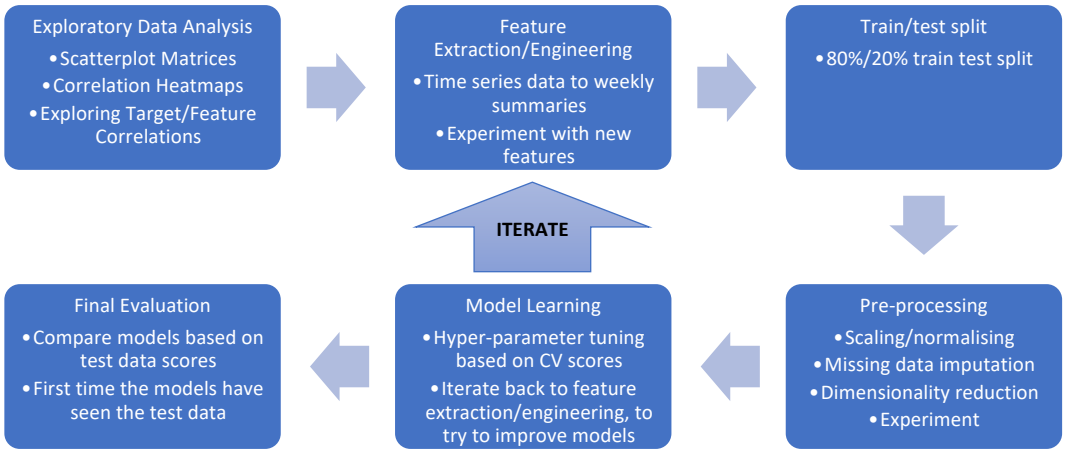
After initial feature extraction/engineering we performed a model search. For each target, we fit an assortment of 10 different machine learning models with no hyper-parameter/model tuning (see page 16). This was performed to provide us with an idea as to which model classes may be worth investigating as well as provide a benchmark for model performance.

Table 1: Cross-validated Model Search Results for Flourishing Scale Classification

model_flourishing_scale_imp_class_post	CV_score_neg_log_loss	std
<class 'xgboost.sklearn.XGBClassifier'>	-0.596505	0.183204
<class 'sklearn.linear_model.logistic.Logistic...	-0.673362	0.411111
<class 'sklearn.ensemble.forest.RandomForestCl...	-0.674880	0.142415
<class 'sklearn.gaussian_process.gpc.GaussianP...	-0.693147	0.000000
<class 'sklearn.neighbors.classification.KNeig...	-0.726639	0.150441
<class 'sklearn.svm.classes.SVC'>	-0.748413	0.055526
<class 'sklearn.neural_network.multilayer_perc...	-0.776776	0.466596
<class 'sklearn.ensemble.weight_boosting.AdaBo...	-2.057285	2.577437
<class 'sklearn.tree.tree.DecisionTreeClassifi...	-9.498323	8.146085
<class 'sklearn.naive_bayes.GaussianNB'>	-11.365097	6.703278
<class 'sklearn.discriminant_analysis.Quadrati...	-12.952121	6.961563

Based on the model search as well desiring heterogenous models (we are more likely to find a higher performing model by investigating different classes of machine learning models), we settled on Regularised Generalised Linear Models (GLM), Support Vector Machines (SVM), Gradient Boosting Machines (GBM) and K-nearest Neighbours (KNN). Using the machine learning cycle summarised in the figure below, each of the four machine learning models were investigated and tuned in depth in order to create the best performing model for each class.

Figure 4: Our Machine Learning Cycle



3.2 Feature Extraction and Engineering

This is discussed in more detail in the Data section on page 2.

3.3 Train-Test Split and Cross Validation

In order to assess model generalisability, it is important to set aside a portion of the data (the test data) that is unseen by the models during the learning phase and only used in the final model evaluation. A training/test split of 80%/20% was used. When tuning the models, 5 to 10-fold cross-validation (CV) was used (Hastie, Tibshirani, & Friedman, 2016, p. 242) to estimate the out of sample performance. In the training data, some observations had missing values for some of their targets. To ensure our test dataset had no missing targets, we sub-sampled from the observations that had no missing target values for the test dataset. Whilst this may introduce some bias in the test dataset towards those that have no missing targets, we believe this is worth the trade-off for a unified set of test data for all targets.

3.4 Pre-processing

Scaling – For the distance-based models (and PCA), failure to scale the features will bias the model towards features of larger scales (Muller & Guide, 2016, p. 135). We used z-score scaling (dividing by the standard deviation and subtracting the mean) on the features for the distance-based models. Z-score scaling was preferred over other scaling methods as it links back to l_2 (Euclidean) norms better than other forms of scaling such as min-max scaling (Bergstra & Bengio, 2012).

Dimensionality reduction and feature selection – We were cognisant of the "curse of dimensionality" (James, Witten, Hastie, & Tibshiranie, 2013, p. 108) and nuisance features (Hastie, Tibshirani, & Friedman, 2016, p. 431) impacting model performance for the KNN and SVM models respectively. The remedy for this was using PCA, a commonly used dimensionality reduction method that distils the features down to smaller number of orthogonal principal components that explain most of the variance (Muller & Guide, 2016). The proportion of total variance explained by the principal components was treated as a hyper parameter randomly grid searched across.

For all our models, we also tested out using differing subsets of the features (all, week 10 only, weeks 9-10 only) as it is possible that the weekly summarises towards the end of the term will be the most important for the post-term scores.

Missing data imputation of training targets – We experimented with models that used KNN to complete partial survey results to expand the training data set. The KNN-imputation used 5 neighbours to complete surveys before the target score was calculated in the training dataset. We treated using either the raw data (with incomplete observations excluded) or data with imputation as a hyper-parameter to tune. We stress that we did not impute any targets on the test dataset (which would be incorrect), we imputed targets on the training data only. This does not amount to target leakage; it is simply the salvaging of incomplete training observations.

3.5 Model Learning

Hyper-parameter Tuning – We have generally performed randomised grid searches as they typically outperform (for the same amount of compute time) and is more efficient than a brute-force grid search (Bergstra & Bengio, 2012).

3.6 Metrics

Classification – Log loss is a commonly used classification metric and was selected as the primary model evaluation metric. Log loss factors in the certainty (e.g. predicted probability) of model predictions. ROC AUC, a metric most concerned with the ordering/ranking of observations was also considered. Factors such as accuracy were disregarded as they do not factor in the sensitivity/recall trade-off (Muller & Guide, 2016, p. 283).

Regression – We use Mean Squared Error (MSE) to compare models. MSE summarises the general misfit of the model using an l_2 (Euclidean) penalty. The l_2 penalty is particular sensitive to outliers. The MSE metric is appropriate when either no outliers exist or you are particularly interested in modelling outliers correctly. MSE is appropriate for modelling the PANAS positive, PANAS negative and flourishing scale survey results as Figure 1 shows that we do not have any outlier observations, all observations are of a similar magnitude and we want to harshly penalise predictions that are particularly far away from the true prediction.

Diagnostics – Whilst log loss and MSE are useful for ranking and comparing models, they do not provide insight into why a model provided a good or bad fit. In this regard, we also examined Predicted vs. Observed by quintiles as well as the Lorenz curve to give us a visual indication of model performance. Whilst the GLMs and tree-based models have in built measures of feature importance (regression coefficients, β and feature importance based on node impurity), KNN and SVM have no such luxuries. SHAP values are model agnostic, consistent and have good theoretical groundings in game theory (Lundberg & Lee, 2017). Thus, we have opted to use SHAP feature importance and SHAP dependence plots to understand ours models better.

3.7 Model 1: Regularised GLMs

Overview of Method – We consider a class of elastic-net generalized linear models (GLMs). Elastic-net GLMs have three main components, a likelihood function describing the distribution of the data, a set of coefficients to describe a linear relationship between the predictors and the response and a penalty to regularise the GLM parameters. All GLM models were fitted using h2o (H2O, 2019).

3.7.1 Feature Selection and Pre-Processors

Feature Subset Selection – Feature selection is performed in the optimisation process for a GLM with an elastic net penalty term. The elastic net penalty will remove variables by setting the coefficients of some features to 0. Elastic net GLMs work well in cases where we have more features than observations (Hastie, Tibshirani, & Friedman, 2016). We also ran a grid search over the feature subsets of all features, week 10 features or week 9-10 features. For simplicity we do not consider feature interactions.

Pre-processors: Imputation – All missing data in the observations is dealt with using mean imputation. This has the interpretation that missing data does not contribute information around variation from the mean response.

Pre-processors: Scaling – All features (except for the intercept term) are normalised using the z-score transformation $z = \frac{x-\bar{x}}{\hat{\sigma}}$. This normalisation ensures that each feature has the same weighting when calculating the regularisation penalty term described below.

3.7.2 Hyper-parameter

For the elastic net GLMs we consider 2 sets of hyper parameters; the regularisation hyper-parameters used to control the size and sparsity of regression coefficients, and GLM family parameters used to provide a hypothesis for the statistical distribution of the response. We used a cross-validation split of 5, since H2O produced an error relating to lack of data when attempting to use 10 splits.

Regularisation Hyper-Parameters – The Elastic net GLM is a generalisation and combination of both LASSO (l_1) regularisation and ridge (l_2) regularisation. In H2O the error term is parameterised as $\lambda\left(\alpha||\beta|\big|_1 + \frac{1-\alpha}{2}||\beta||_2^2\right)$, for $\alpha \in [0,1]$ and $\lambda \geq 0$, where β are the GLM coefficients. Elastic net penalty has advantages over both Ridge regression and LASSO regression; the l_1 penalty term creates a sparse model (ridge regression doesn't do this) and the l_2 penalty term assists in identifying impacts of multiple correlated features exist (LASSO regression will generally only keep one of multiple correlated features).

H2O uses efficient algorithm to search all along a path for λ (where α is held constant), at a similar computation cost as only evaluating the function at a single (α, λ) point (Friedman, Hastie, & Tibshirani, 2010). This allows us to search the 2-D grid of (α, λ) by only specifying a few points of α. The search space includes LASSO $(\alpha = 1)$ and ridge regression $(\alpha = 0)$ as special cases.

GLM Family Hyper-Parameters – For regression tasks we consider Gaussian, Tweedie, Gamma, Poisson and Negative Binomial families with their respective canonical link functions. For classification problems only the logistic link Binomial family is considered. The Tweedie and Negative Binomial distributions require an additional parameter to define the relationship between the mean and variance. In the Tweedie distribution (where $Var(Y) = \sigma^2\mu^p$) we search for the Tweedie variance power (p) values between 1 and 2. For Negative Binomial (where $Var(Y) = \mu + \theta\mu^2$) we tune θ over positive real numbers.

Tuning – Each family was optimised using a 2-D grid search (over α and feature weeks) or 3-D grid search for Tweedie and Negative Binomial Family only (2-D search and the variance parameter). A comprehensive grid search is feasible due to the low dimensionality of the hyper-parameter space and small data set.

3.8 Model 2: Support Vector Machines

Overview of Method – Support Vector Machines are strong, distanced based learners that learn by finding separating hyperplanes. Whilst hyperplanes are linear in nature, the "kernel trick" allows the efficient projection of data and fitting of a separating hyperplane in a higher dimensional space, effectively allowing non-linearity to be handled (James, Witten, Hastie, & Tibshiranie, 2013, pp. 337-355). The models were implemented in scikit-learn (scikit-learn SVM documentation, 2019) and the results of this random grid search are shown in the appendices on page 20. Given the small size of our dataset, we have the luxury of plentiful compute; thus, we have treated the level of dimensionality reduction (via PCA), the pre-processing method, whether we used the target imputation (see page 5) as hyper-parameters that we randomly grid searched across alongside our SVM hyper-parameters.

3.8.1 Feature Selection and Pre-Processors

Feature Subset Selection – We considered the following subsets of features for our SVMs: all features, week 9-10 features and week 10 features. As discussed on page 4, nuisance/noise parameters which have no signal can impact model performance. Which subset of features that fed into the model was treated as a hyper-parameter.

Pre-processors: Imputation – We considered the following simple imputation strategies: most frequent value, mean and median values. These were treated as a hyper-parameter and randomly grid searched across. Whether we used imputed target for the training data or not was treated as a hyperparameter[1].

Pre-processors: Scaling and Principal Component Analysis – Discussed on page 4.

3.8.2 SVM Hyper-parameters

The SVM hyper-parameters are documented in (scikit-learn SVM documentation, 2019)

C – penalty parameter for the error term – A regularization parameter. A soft-margin support vector classifier/regressor has a "budget" for error/slack of the separating hyperplane (Hastie, Tibshirani, & Friedman, 2016, p. 419). The hyper-parameter C is the penalty for the total error/slack values when fitting. Values of between 0.01 and 1000 were randomly grid searched across.

Kernel – Kernel function used for the "kernel trick", allowing efficient searching of separating hyperplanes in higher dimensional spaces. This is an important hyper-parameter to search across as it effectively allows us to try separating the classes with various "shapes". Linear, polynomial and radial basis function kernels (RBF) were fitted.

Gamma – RBF kernel parameter that sets the influence of a single training observation and how "closely" the model fits to the data. Gamma values of between 0.01 and 100 were fitted.

Degree – This represents the degree of the polynomial, when a polynomial kernel is used. Degrees of two to six were fitted.

3.9 Method 3: K-Nearest Neighbours

Overview of Method – The k-nearest neighbour (KNN) algorithm is a distance-based machine learning algorithm. KNN takes the majority vote and mean of the k-nearest neighbours of the observation we would like to predict within the training data for classification and regression respectively. Each feature is treated as a dimension when calculating distance (Dietrich, 1994). Scikit-learn was used to fit the models.

3.9.1 Feature selection and pre-processing

Feature Selection – We considered three feature subsets: all features, week 10 only and weeks 9 and 10 only. This was treated as a hyper-parameter.

Pre-processors: Imputation – The simple imputation strategies of most frequent, mean and median were treated as a hyper-parameter.

Pre-processors Scaling and PCA – see page 7

[1] We stress that we did not impute the target on the test data (which would be incorrect), we imputed the targets for the training data only to extract more signal out of the limited training data available.

3.9.2 Hyper-parameters

N – The number of neighbours and the namesake of the model. This is the hyperparameter that determines the number of closest neighbours that are used for prediction. Values of 1-25 were searched across.

Weights – When taking the majority vote or average, we can either give a uniform weight to the k nearest neighbours (e.g. simple average/vote) or weight the averages/votes by the inverse of the distance (e.g. closer neighbours are weighted higher).

Metric – Whether to use l_1 (Manhattan) or l_2 (Euclidean) distance when calculating distance.

3.10 Method 4: Ensembles of Trees

3.10.1 Overview of Method

For the ensemble learning methods, we investigated gradient boosted decision trees with the XGBoost implementation. Gradient boosting works by iteratively creating an ensemble of weak learners (e.g. decision tree "stumps") to gradually minimise the loss function at a specific learning rate (Guestrin & Chen, 2016). A cross-validated randomised grid search was carried out to find the optimal GBM model for each target.

Pre-processors: Imputation – the default XGBoost missing data imputation method was used. The XGBoost package imputes data by considering the missing data assignments at each node that will lead to the greatest reduction in the cost function (Guestrin & Chen, 2016).

Pre-processors: Scaling and Dimensionality Reduction – as XGBoost is a tree-based model, scaling and dimensionality reduction is not required.

3.10.2 XGBoost Hyper-parameters

The random grid search results are listed in the appendix (see page 25).

Learning Rate – This controls the shrinkage rate of the residual in each split (XGBoost Documentation, 2019). Higher learning rates can be computationally quicker but may "descend" down the cost function too quickly leading to over-fitting or overshooting the global cost function minimum, landing at a local minimum. A lower learning rate allows for slower adaption to the data, potentially avoiding these problems. Given the small size of the dataset, we can afford the extra compute time potentially required by having a lower learning rate and have performed our randomised grid search over lower learning rate values.

Gamma – this is also known as the minimum split loss, a regularisation parameter that restricts the split at the leaf node of the tree – the tree will be pruned if the loss reduction threshold is not met (XGBoost Documentation, 2019). It is closely related to the max tree depth hyper-parameter. If the max depth of tree is large, the gamma must be set high enough to stop the trees from growing too deeply and overfitting.

Max depth – the maximum depth that a tree can have. High max depth will contribute to a complex model that is vulnerable to overfitting as it captures noise from the training data making it unable to generalise to unseen data.

Column sample by tree – the number of features used to train each tree. Can be used to reduce dimensionality.

Subsample – this is the ratio of training data that will be used to grow the tree in each boosting round and can be used to prevent the model from overfitting the full set of training data.

Min child weight – a regularisation parameter that stops the tree from partitioning further if the number of instances in a leaf node does not meet the minimum requirement, effectively limiting the tree depth.

Alpha – this is an l_1 regularisation term that helps penalize the l_2 loss function so that the weights will not overfit the training set and lead to a higher bias. A higher value of alpha would make the model more robust to overfitting.

Lambda – similar to Alpha but used to control the l_2 loss function.

3.11 Method 5: Baseline Model

As a sense check for all models, the results were compared to the simplest estimator possible, the mean of all observations in the training data set. Given the low number of data points this baseline assists in identifying cases where a model has over-fit the training data set.

4 Results

4.1 Flourishing Score Classification

We found that the Regularised GLM provided the best classification results for on the test data set. This was closely followed by the Support Vector Machine. Neither the XGBoost model or the K nearest neighbours' method was able to provide any uplift in predictive power over the benchmark model.

Table 2: Flourishing Score Classification Model Rankings (Log Loss)

Rank (Test Score)	Model	Test Score	Training CV Score (mean)	Training CV Score (standard deviation)
1	Regularised GLM	0.6460	1.0125	1.0472
2	Support Vector Machine	0.6638	0.5103	0.1427
3	Mean (benchmark model)	0.6931	NA	NA
4	Gradient Boosting (XGBoost)	0.7197	0.4975	0.2563
5	KNN	0.8226	0.6061	0.2419

The best GLM model was a Binomial GLM model, using only week 10 data with alpha = 0.1 and lambda = 0.1999. This gave a total of 24 active predictors.

For a regularised GLM (with scaled predictors) the feature importance is simply the magnitude of the coefficient. This coefficient is also the SHAP value for the predictor. We found that the most predictive feature is the proportion of time a person spends running, where a higher proportion of running time is related to having a higher flourishing score that is above the median. This suggests that this model will predict all participants who have a proportion of running time higher/lower than the average will also have flourishing score higher/lower than the median. Other important features are the average speed (higher speed, which is likely related to more movement, gives a higher probability of having a high flourishing score. Outdoors distant move is correlated with a lower probability of having a high flourishing score. Finally, the proportion of time a person is in a silent/noisy environment is correlated with having a lower/higher probability of having a high flourishing score.

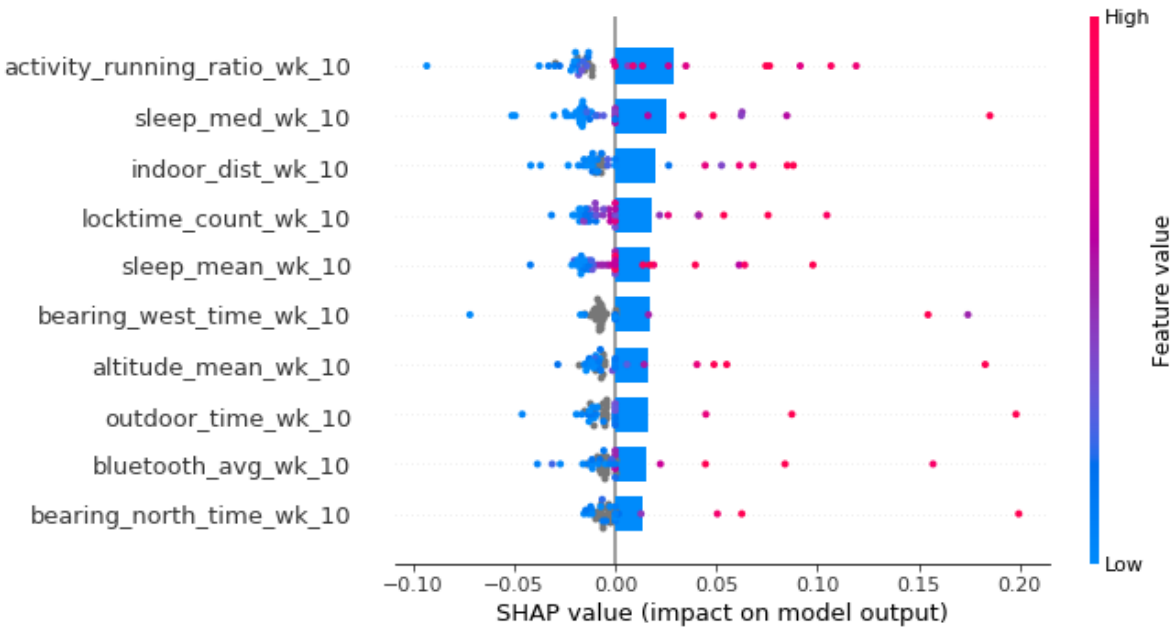
Table 3: Flourishing Score Classification GLM Table of coefficients

Feature	Coefficient	Feature	Coefficient

Intercept	-1.6029		
activity_running_ratio_wk_10	19.9665	**speed_max_wk_10**	0.0141
speed_mean_wk_10	2.0505	**sleep_mean_wk_10**	0.0125
outdoors_dist_wk_10	-1.0557	**altitude_max_wk_10**	0.0033
audio_silent_ratio_wk_10	-0.7366	**bearing_west_time_wk_10**	0.0021
audio_noisy_ratio_wk_10	0.2613	**location_3_time_wk_10**	-0.0003
speed_sd_wk_10	0.2084	**location_1_time_wk_10**	0.0002
sleep_med_wk_10	0.0450	**location_5_time_wk_10**	0.0001
altitude_mean_wk_10	0.0358	**location_2_time_wk_10**	0.0001
locktime_count_wk_10	0.0231	**travelstate_time_stationary_wk_10**	0.0000
chargetime_count_wk_10	0.0187	**indoor_time_wk_10**	0.0000
altitude_sd_wk_10	0.0186	**chargetime_q3_wk_10**	0.0000
sleep_max_wk_10	0.0148	**locktime_q1_wk_10**	0.0000

These features are similar to the import features identified in the SVM model (the second best performing model) as shown in the SHAP plot below.

Figure 5: SVM SHAP values



Additional diagnostic plots are in the appendix on page 30.

4.2 Flourishing Score Regression

Table 4: Flourishing Score Regression Model Rankings (Mean Squared Error)

Rank (Test Score)	Model	Test Score	Training CV Score (mean)	Training CV Score (standard deviation)
1	Mean (benchmark model)	51.0240	NA	NA
2	Regularised GLM[2]	51.0240	119.7043	122.5224
3	KNN	52.8751	67.9234	64.1468
4	Gradient Boosting (XGBoost)	57.6100	89.4232	60.3546
5	Support Vector Machine	71.3403	76.7164	74.4157

[2] Cross-validation found the best GLM model was a constant (intercept only) model. This is identical to the benchmark model.

We were not able to find any model that beat the baseline (mean) model. This finding agrees with the findings in (Wang, et al., 2014) which do not find any activity sensor data is correlated with both pre and post Flourishing scores and only the number of Bluetooth co-locations is weakly correlated with the post flourishing score. The difference between our results and the results in (Wang, et al., 2014) can likely be attributed due to our exclusion of data to create a train/validate/test split.

4.3 PANAS Positive Classification

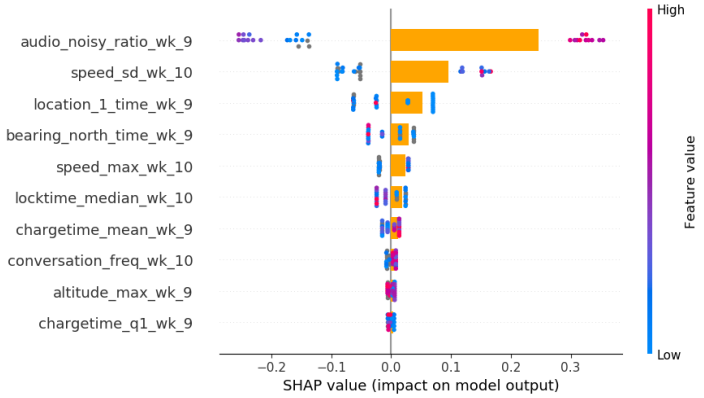
For the PANAS score positive classification, the XGBoost classifier performed the best on the test set, followed by KNN, SVM, the benchmark model and finally the regularised GLM. The XGBoost classifier also had the lowest CV standard deviation (implying a lower variance estimator).

Table 5: PANAS Positive Classification Model Rankings (Log Loss)

Rank (Test Score)	Model	Test Score	Training CV Score (mean)	Training CV Score (standard deviation)
1	Gradient Boosting (XGBoost)	0.6346	0.6540	0.1164
2	KNN	0.6667	0.6290	0.1470
3	Support Vector Machine	0.6931	0.5651	0.1359
4	Mean (benchmark model)	0.7052	NA	NA
5	Regularised GLM	0.7557	2.0445	2.8246

The best performing XGBoost model used 500 trees, with the best performing model using a max depth of 7. Relative to the size of the dataset, the max depth and number of trees is a little high (although this is offset somewhat by the lower learning rate), so we were cognisant of overfitting. However, the test score is not much lower than the training score suggesting that we have not terribly overfit to the data.

Figure 6: XGBoost SHAP values



Audio_noisy_ratio has the largest feature importance for the XGBoost model, followed by speed_sd_wk_10 and location_1_time_wk_9. The model suggests that the level of noise (indicative of social activity) as well as movement (indicated by speed) is linked to PANAS positive score.

4.4 PANAS Positive Regression

Table 6: PANAS Positive Regression Model Rankings (Mean Squared Error)

Rank (Test Score)	Model	Test Score	Training CV Score (mean)	Training CV Score (standard deviation)

1	Mean (benchmark model)	37.8894	NA	NA
1	Regularised GLM[3]	37.8894	86.6610	119.9731
3	Support Vector Machine	39.3396	41.7551	46.2614
4	Gradient Boosting (XGBoost)	40.3416	34.0923	16.2509
5	KNN	44.64	36.7413	22.3101

We were not able to find any model that beat the baseline (mean) model. This finding agrees with the findings in (Wang, et al., 2014) do not find any activity sensor data is correlated either pre or post positive PANAS scores. (Wang, et al., 2014) Only report that the photographic affect meter results have a correlation with Positive PANAS, but we did not include that features in our analysis.

4.5 PANAS Negative Classification

The Support Vector Machine had the best performance on the test data, followed closely by baseline (mean) model, gradient boosting and KNN. This distance-based models performed the best on the test dataset, with the more powerful SVMs (when compared to KNN) being more robust to nuisance features as well as being able to discriminate between more and less important features (Kouiroukidis & Evangelidis, 2011).

Table 7: PANAS Negative Classification Model Rankings (Log Loss)

Rank (Test Score)	Model	Test Score	Training CV Score (mean)	Training CV Score (standard deviation)
1	Support Vector Machine	0.6306	0.4587	0.2112
2	Mean (benchmark model)	0.6872	NA	NA
3	Gradient Boosting (XGBoost)	0.7797	0.6000	0.1617
4	KNN	0.7993	0.5918	0.2246
5	Regularised GLM	0.8559	0.7491	0.3648

The best fitting SVMs had a regularisation parameter C of 1, used enough principal components to explain 60-70% of the feature variance (suggesting some noise in the data), and RBF kernels. Whilst the Support Vector Machine performed the best on the training dataset, performance was significantly worse on the test data than the training data implying that the model has overfit heavily. The Gini index is negative (implying predictions that are worse than random) on the test set and the predicted vs. observed charts show an inability to rank observations (see page 32). On this basis, we conclude that the mean (benchmark) model is the best model for this target and we are unable to find an effective machine learning model for this target. This finding agrees with the findings of (Wang, et al., 2014) who not report any features (from the activity sensor data or otherwise) is correlated either pre or post negative PANAS scores.

4.6 PANAS Negative Regression

Table 8: PANAS Negative Regression Model Rankings (Mean Squared Error)

Rank (Test Score)	Model	Test Score	Training CV Score (mean)	Training CV Score (standard deviation)
1	Mean (benchmark model)	45.2989	NA	NA
2	Regularised GLM	51.3832	76.9620	32.7055
3	Support Vector Machine	61.4188	44.9785	25.2538
4	KNN	81.2133	46.857	16.4811
5	Gradient Boosting (XGBoost)	82.3442	39.6672	14.4794

Similar to PANAS Negative Classification, we were not able to find any model that beat the baseline (mean) model.

[3] Cross-validation found the best GLM model was a constant (intercept only) model. This is identical to the benchmark model.

5 Discussion

The prediction of mental health survey results in the StudentLife data set is a challenging problem. The low number of observations and high data dimensionality provides a challenge for many algorithms as they either suffer from the curse of dimensionality, do not include an inbuilt feature selection algorithm or typically require large data sets to provide accurate predictions. Additionally, there was a material amount of missing data in both the raw activity log and the features used for prediction. The missing data was most prominent in the later weeks of the observation period which impacted features that would otherwise be useful.

5.1 Comparison of performance between methods

We found that simple models consistently out-performed more complex models across all tasks. For all three regression tasks the simplest model (a constant prediction) was the best model. This suggests that there is no linear or non-linear relationship between the predictors and the physiological survey results.

In classification models we were able to produce models that could discriminate between the two classes and a different model was the best model for each of the three classification tasks. There is a large amount of noise in the tests data set predictions – for example the PANAS negative classification model performs best in MSE but very poorly on other auxiliary metrics.

We found that features in later weeks were more included in the both the flourishing score and PANAS positive classification models. This suggests a strong temporal component to physiological scores. The most important features in the flourishing score and PANAS positive classification were related to moved (running / speed features) and sound (noisy/quiet environments). We found that more time in noisy environments (likely due to conversation or other fun activities) and more movement resulted in more positive mental health scores.

5.2 Discussion of advantages and disadvantages of various methods

Table 9 below, summarises the relative advantages and disadvantages we found when fitting various models. The discussion is focused on advantages and disadvantages to the problem at hand where dealing with small data sets, identifying important features, handling missing data and fitting non-linear features is important.

Table 9: Comparison of implemented methods

Method	Advantages	Disadvantages
GLM	• Easy to interpret (read off coefficients) • Automatically finds a sparse model in small data where the feature matrix is not full rank. • Provides a probabilistic interpretation of predictions	• Needs additional feature engineering to handle interactions, non-linear features and missing values.
SVM	• Kernels allow identification of non-linear features • Tuning allows simple models to be identified	• Does not include inbuilt feature selection • Performance deteriorates in the presence of noise features • Needs additional tools (e.g. SHAP) to explain and interpret important features and impacts
KNN	• Can identify simply models that generalise well to new data • Fast to train and iterate over hyper-parameters	• Does not include inbuilt feature selection • Suffers heavily from the “curse of dimensionality • Needs additional tools to explain and interpret important features and impacts
XGBoost	• With extensive tuning XGBoost was able to produce non-linear models • Automatically fits non-linear features and feature interactions • Naively treats missing features (as a different level to split a tree on)	• Prone to over-fitting. A large amount of time was spent tuning the hyper-parameters to simplify the model to get reasonable validation performance • Performance is very dependent on hyper-parameter selection • XGBoost can be biased if a small number of trees is used

	• Includes features selection as part of the model fitting algorithm (what features to split on)	

5.3 Discussion of findings based on results

Predicting mental health outcomes correctly is a challenging problem. Many models either over-fit or do not generalise to new data. This is particularly evident in the large number of models that either do not beat the baseline model or perform very poorly on the test dataset. A large contributing factor to this is the very small data size and many methods applied were found to not be appropriate. Models without features selection (KNN, SVM) generally performed poorly due to the high number of nuisance parameters encountered.

The classification problem appears to be easier to solve than any of the regression problems considered. We may be able to attribute a part of this finding to the fact that binning the data may remove some of the noise and reduce the complexity of the raw survey scores.

We found that features relating the audio noise and movement are the most predictive factors for having a high flourishing score or high positive PANAS score. Participants spending more time in noisy environments (likely engaging in conversation or other social activity) and participants with more movement are likely to achieve higher flourishing scale and positive PANAS scores. We conclude that tracking sound levels and user activity may have some benefit in identifying participants that are likely to have strong of moderate-low mental health flourishing scale or PANAS positive scores. Translating the physiological surveys into identifying at risk individuals was not considered in this analysis but would be a direction for future research.

More research would be required before any tool could be developed from these models. In particular the high level of uncertainty in the cross-validation scores suggests we do not have enough data to draw statistically significant conclusions. While we were able to improve on a baseline model for classifying scores as above or below the median score – translating this into clinical action (where very low scoring outliers are the primary concern) requires additional research.

5.3.1 Comparison of our findings to (Wang, et al., 2014)

For positive and negative PANAS scores the superior test data set performance of the mean model over more complex statistical techniques (including linear regression) agrees with the authors of (Wang, et al., 2014). (Wang, et al., 2014) do not report any factors in the activity log data having a linear relationship with either positive or negative PANAS scores. Our finding that the baseline model is the best model for the PANAS regression problems provides further evidence that the activity log data cannot be used to predict raw PANAS positive or negative scores.

For the Flourishing Scale the authors of (Wang, et al., 2014) only report one automatic sensor data feature (the number of co-locations) that has a significant correlation to the post observation participant score. The authors report a further two features (conversation duration and conversation duration during evening) that have a significant correlation with the pre observation score. The inclusion of the audio features in our flourishing scale classification model agrees with this insight. The different features between our best (GLM) model and the correlation analysis of (Wang, et al., 2014) may be caused by splitting the full data set into a training and test split, as this removes a large number of observations.

The correlation analysis of Wang, et al is not statistically sound. The authors appear to conduct over 100 significance tests, without stating a clear hypothesis or making any allowance for multiple hypothesis testing. The authors report multiple values with low p-values however the threshold they use to report p-values appears to change between tables in the report without a clear explanation. Additionally, for many of the correlations report, the feature is only significant for either pre or post survey results (but not both) and no clear explanation for this is provided.

6 Conclusion

We have comprehensively demonstrated that machine learning only adds marginal value over basic statistical methods such as linear correlation analysis in limited cases and in most cases adds no or negative value. Applying machine learning to the StudentLife dataset post observation scores is analogous to cracking a walnut with a sledge hammer. The classification examples where we did improve on the baseline model are likely of limited use as our classification relates to the median score, but in practice we are likely interested in finding particular low observations for individuals who are at risk of depression.

Our conclusion agrees with other work on using AI to predict social outcomes (in this case depression) either does not work or doesn't perform better than simple linear predictions (Narayana, 2019). Many recent break-throughs in machine learning have come from cases using extremely large datasets. The StudentLife dataset is a small (but wide) dataset and the successes of methods with large datasets cannot be assumed to carry over into this setting. Even in cases of moderate dataset sizes with (in the range of 10,000 – 1,000,000 observations) machine learning does not improve on domain knowledge and (generalised) linear regression using one or two factors. This recent nature articles (DeVries, Viegas, Wattenberg, & Meade, 2018) and (Mignan & Broccardo, 2019) demonstrate the danger of apply machine learning without referencing a baseline model.

Based on our findings, we must conclude that predicting student mental health outcomes using the automatic sensing data in the StudentLife dataset this is not an appropriate task for machine learning. In all regression cases we found that no model was able to improve on a simple baseline of the mean only. These cases can be interested as that we did not find any factors are useful in predicting student mental health outcomes. In additional where we were able to beat the baseline, the improve model was generally a very simple model. We conclude that in this setting machine learning and automated sensing data is not a substitute for domain knowledge and clinical work.

7 Appendix

7.1 Exploratory Data Analysis

Figure 7: Flourishing Scale Correlation Heatmap

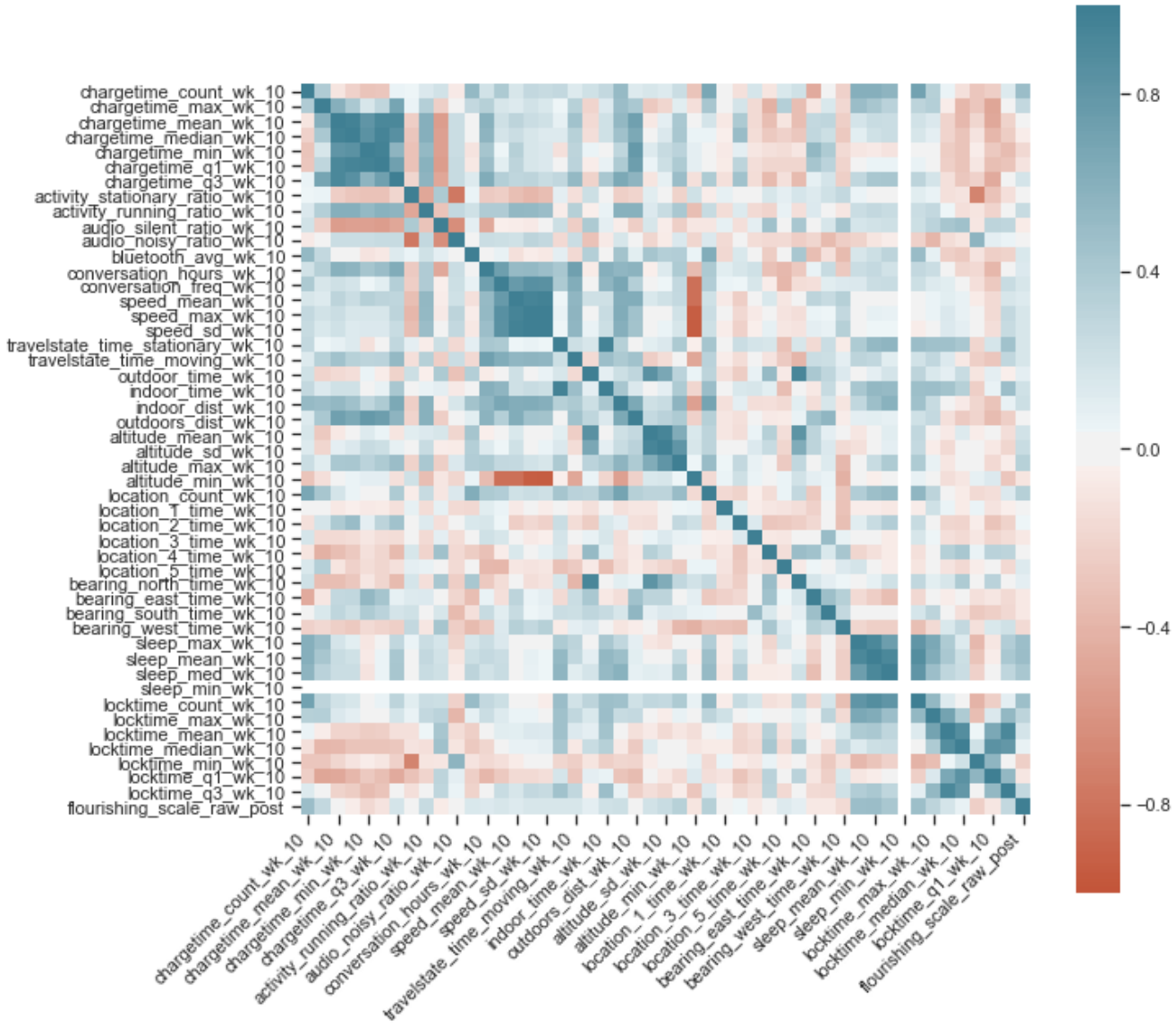


Figure 8: PANAS Positive Score Correlation Heatmap

Figure 9: PANAS Negative Score Correlation Heatmap

7.2 Model Search

Table 10: Flourishing Scale Classification Model Search

	model_flourishing_scale_imp_class_post	CV_score_neg_log_loss	std
9	<class 'xgboost.sklearn.XGBClassifier'>	-0.596505	0.183204
10	<class 'sklearn.linear_model.logistic.Logistic...	-0.673362	0.411111
0	<class 'sklearn.ensemble.forest.RandomForestCl...	-0.674880	0.142415
3	<class 'sklearn.gaussian_process.gpc.GaussianP...	-0.693147	0.000000
1	<class 'sklearn.neighbors.classification.KNeig...	-0.726639	0.150441
2	<class 'sklearn.svm.classes.SVC'>	-0.748413	0.055526
8	<class 'sklearn.neural_network.multilayer_perc...	-0.776776	0.466596
5	<class 'sklearn.ensemble.weight_boosting.AdaBo...	-2.057285	2.577437
4	<class 'sklearn.tree.tree.DecisionTreeClassifi...	-9.498323	8.146085
6	<class 'sklearn.naive_bayes.GaussianNB'>	-11.365097	6.703278
7	<class 'sklearn.discriminant_analysis.Quadrati...	-12.952121	6.961563

Table 11: Flourishing Scale Regression Model Search

	model_flourishing_scale_imp_post	CV_score_neg_mean_squared_error	std
1	<class 'sklearn.neighbors.regression.KNeighbor...	-79.763333	66.120698
5	<class 'sklearn.ensemble.weight_boosting.AdaBo...	-82.787992	83.986742
2	<class 'sklearn.svm.classes.SVR'>	-85.467658	85.961592
0	<class 'sklearn.ensemble.forest.RandomForestRe...	-94.167372	79.190334
7	<class 'xgboost.sklearn.XGBRegressor'>	-117.356846	102.107234
4	<class 'sklearn.tree.tree.DecisionTreeRegressor'>	-177.800000	130.655969
6	<class 'sklearn.neural_network.multilayer_perc...	-1091.262682	360.372214
3	<class 'sklearn.gaussian_process.gpr.GaussianP...	-1990.333333	377.412735

Table 12: PANAS Positive Classification Model Search

	model_panas_pos_imp_class_post	CV_score_neg_mean_squared_error	std
6	<class 'sklearn.naive_bayes.GaussianNB'>	-0.400000	0.181812
10	<class 'sklearn.linear_model.logistic.Logistic...	-0.400000	0.364768
8	<class 'sklearn.neural_network.multilayer_perc...	-0.441667	0.329246
7	<class 'sklearn.discriminant_analysis.Quadrati...	-0.450000	0.256038
5	<class 'sklearn.ensemble.weight_boosting.AdaBo...	-0.466667	0.355903
1	<class 'sklearn.neighbors.classification.KNeig...	-0.491667	0.191667
0	<class 'sklearn.ensemble.forest.RandomForestCl...	-0.500000	0.378227
4	<class 'sklearn.tree.tree.DecisionTreeClassifi...	-0.508333	0.155680
2	<class 'sklearn.svm.classes.SVC'>	-0.575000	0.237024
9	<class 'xgboost.sklearn.XGBClassifier'>	-0.633333	0.244949
3	<class 'sklearn.gaussian_process.gpc.GaussianP...	-0.658333	0.294510

Table 13: PANAS Positive Regression Model Search

	model_panas_pos_imp_post	CV_score_neg_mean_squared_error	std
0	<class 'sklearn.ensemble.forest.RandomForestRe...	-50.358410	22.188271
7	<class 'xgboost.sklearn.XGBRegressor'>	-54.176824	25.905802
5	<class 'sklearn.ensemble.weight_boosting.AdaBo...	-56.117604	22.744026
2	<class 'sklearn.svm.classes.SVR'>	-56.648800	24.014415
1	<class 'sklearn.neighbors.regression.KNeighbor...	-60.038667	32.643276
4	<class 'sklearn.tree.tree.DecisionTreeRegressor'>	-85.133333	55.242938
6	<class 'sklearn.neural_network.multilayer_perc...	-622.745779	230.568387
3	<class 'sklearn.gaussian_process.gpr.GaussianP...	-921.933333	232.915855

Table 14: PANAS Negative Classification Model Search

	model_panas_neg_imp_class_post	CV_score_neg_mean_squared_error	std
5	<class 'sklearn.ensemble.weight_boosting.AdaBo...	-0.350000	0.265623
6	<class 'sklearn.naive_bayes.GaussianNB'>	-0.408333	0.284922
4	<class 'sklearn.tree.tree.DecisionTreeClassifi...	-0.416667	0.368932
2	<class 'sklearn.svm.classes.SVC'>	-0.433333	0.081650
9	<class 'xgboost.sklearn.XGBClassifier'>	-0.433333	0.280872
1	<class 'sklearn.neighbors.classification.KNeig...	-0.458333	0.250693
0	<class 'sklearn.ensemble.forest.RandomForestCl...	-0.475000	0.129368
7	<class 'sklearn.discriminant_analysis.Quadrati...	-0.491667	0.205649
3	<class 'sklearn.gaussian_process.gpc.GaussianP...	-0.558333	0.197379
8	<class 'sklearn.neural_network.multilayer_perc...	-0.558333	0.190212
10	<class 'sklearn.linear_model.logistic.Logistic...	-0.575000	0.330088

Table 15: PANAS Negative Regression Model Search

	model_panas_neg_imp_post	CV_score_neg_mean_squared_error	std
0	<class 'sklearn.ensemble.forest.RandomForestRe...	-59.002427	34.913466
7	<class 'xgboost.sklearn.XGBRegressor'>	-61.583731	41.763242
5	<class 'sklearn.ensemble.weight_boosting.AdaBo...	-62.880365	46.313765
2	<class 'sklearn.svm.classes.SVR'>	-72.726779	47.475281
1	<class 'sklearn.neighbors.regression.KNeighbor...	-77.729333	57.431202
4	<class 'sklearn.tree.tree.DecisionTreeRegressor'>	-142.500000	58.329571
6	<class 'sklearn.neural_network.multilayer_perc...	-449.897426	236.725593
3	<class 'sklearn.gaussian_process.gpr.GaussianP...	-536.700000	201.363378

7.3 Model 1: GLM

Table 16: Flourishing Scale Classification Grid Search Results (only top 15 models shown)

Rank	CV Score (Total)	Imputer Strategy	Feature Subset	GLM Family	Alpha	Best Lambda
1	1.4228	mean (no score imputation)	all	Binomial	0.1	0.2193
2	1.6200	mean (no score imputation)	wk_10	Binomial	0.1	0.1990
3	1.6447	mean (no score imputation)	all	Binomial	0.2	0.1096
4	1.8034	mean (no score imputation)	all	Binomial	0.3	0.0731
5	1.8172	mean (no score imputation)	wk_9-10	Binomial	0.1	0.1990
6	1.9778	mean (no score imputation)	all	Binomial	0.4	0.0548
7	2.0799	mean (no score imputation)	wk_10	Binomial	0.2	0.0995
8	2.1841	mean (no score imputation)	all	Binomial	0.5	0.0439
9	2.2081	mean (no score imputation)	all	Binomial	0.0	0.0219
10	2.2697	mean (no score imputation)	wk_9-10	Binomial	0.2	0.0995
11	2.3780	mean (no score imputation)	all	Binomial	0.6	0.0365
12	2.4471	mean (no score imputation)	all	Binomial	0.7	0.0313
13	2.4792	mean (no score imputation)	all	Binomial	0.8	0.0274
14	2.4880	mean (no score imputation)	wk_10	Binomial	0.3	0.0663
15	2.5060	mean (no score imputation)	all	Binomial	0.9	0.0244

Table 17: Flourishing Scale Regression Grid Search Results (only top 15 models shown)

Rank	CV Score (Total)	Imputer Strategy	Feature Subset	GLM Family	Alpha	Best Lambda	Theta	Tweedie Power
1	102.9487	mean (no score imputation)	all	Tweedie	0.1	52.2755	NA	1.1
2	139.2010	mean (no score imputation)	all	gamma	1.0	0.5228	NA	NA
3	141.5565	mean (no score imputation)	all	Tweedie	0.7	2.4893	NA	1.3
4	143.6438	mean (no score imputation)	all	Tweedie	0.2	26.1377	NA	1.1
5	143.8887	mean (no score imputation)	all	Tweedie	0.3	17.4252	NA	1.1
6	143.9992	mean (no score imputation)	all	Tweedie	0.2	8.7126	NA	1.3
7	144.0291	mean (no score imputation)	all	Tweedie	0.1	17.4252	NA	1.3
8	144.0444	mean (no score imputation)	all	Tweedie	0.4	13.0689	NA	1.1
9	144.0729	mean (no score imputation)	all	Tweedie	0.5	10.4551	NA	1.1
10	144.0911	mean (no score imputation)	all	Tweedie	0.6	8.7126	NA	1.1
11	144.1041	mean (no score imputation)	all	Tweedie	0.7	7.4679	NA	1.1
12	144.1825	mean (no score imputation)	all	Tweedie	0.8	6.5344	NA	1.1
13	144.6301	mean (no score imputation)	all	Tweedie	0.9	5.8084	NA	1.1
14	145.6504	mean (no score imputation)	all	Tweedie	0.9	1.9361	NA	1.3
15	147.3139	mean (no score imputation)	all	Tweedie	0.5	3.4850	NA	1.3

Table 18: PANAS Positive Classification Grid Search Results (only top 15 models shown)

Rank	CV Score (Total)	Imputer Strategy	Feature Subset	GLM Family	Alpha	Best Lambda
1	1.8843	mean (score imputation)	wk_10	Binomial	0.1	0.1638
2	1.9436	mean (no score imputation)	wk_10	Binomial	0.1	0.1593
3	2.0464	mean (score imputation)	wk_10	Binomial	0.2	0.0819
4	2.0867	mean (no score imputation)	wk_10	Binomial	0.2	0.0796
5	2.1838	mean (score imputation)	wk_10	Binomial	0.3	0.0546
6	2.2153	mean (no score imputation)	wk_10	Binomial	0.3	0.0531
7	2.2177	mean (score imputation)	all	Binomial	0.1	0.2834
8	2.2833	mean (no score imputation)	all	Binomial	0.1	0.2834
9	2.3284	mean (score imputation)	wk_10	Binomial	0.4	0.0409
10	2.3315	mean (score imputation)	all	Binomial	0.2	0.1417
11	2.3490	mean (no score imputation)	wk_10	Binomial	0.4	0.0398
12	2.3995	mean (score imputation)	all	Binomial	1.0	0.0283
13	2.4019	mean (score imputation)	all	Binomial	0.3	0.0945
14	2.4025	mean (no score imputation)	all	Binomial	0.2	0.1417
15	2.4391	mean (score imputation)	all	Binomial	0.9	0.0315

Table 19: PANAS Positive Regression Grid Search Results (only top 15 models shown)

Rank	CV Score (Total)	Imputer Strategy	Feature Subset	GLM Family	Alpha	Best Lambda	Theta	Tweedie Power
1	87.9148	mean (no score imputation)	all	Tweedie	0.3	12.4898	NA	1.1
2	87.9215	mean (no score imputation)	all	Tweedie	0.4	9.3673	NA	1.1
3	87.9360	mean (no score imputation)	all	Tweedie	0.2	18.7347	NA	1.1
4	87.9406	mean (no score imputation)	all	Tweedie	0.5	7.4939	NA	1.1
5	87.9609	mean (no score imputation)	all	Tweedie	0.6	6.2449	NA	1.1
6	87.9721	mean (no score imputation)	all	Tweedie	0.1	37.4694	NA	1.1
7	87.9839	mean (no score imputation)	all	Tweedie	0.7	5.3528	NA	1.1
8	88.0106	mean (no score imputation)	all	Tweedie	0.8	4.6837	NA	1.1
9	88.0408	mean (no score imputation)	all	Tweedie	0.9	4.1633	NA	1.1

10	88.4655	mean (no score imputation)	all	Tweedie	1.0	3.7469	NA	1.1
11	95.2398	mean (score imputation)	wk_10	Negative binomial	0.1	2.1392	0.01	NA
12	95.2549	mean (score imputation)	wk_10	Negative binomial	0.1	2.7642	0.0001	NA
13	95.2635	mean (score imputation)	wk_10	Negative binomial	0.1	2.7724	1.00E-08	NA
14	95.2635	mean (score imputation)	wk_10	Negative binomial	0.1	2.7724	1.00E-10	NA
15	95.3015	mean (score imputation)	wk_10	Poisson	0.1	2.7724	NA	NA

Table 20: PANAS Negative Classification Grid Search Results (only top 15 models shown)

Rank	CV Score (Total)	Imputer Strategy	Feature Subset	GLM Family	Alpha	Best Lambda
1	0.8674	mean (no score imputation)	wk_10	Binomial	0.8	0.0253
2	0.8808	mean (no score imputation)	wk_10	Binomial	0.9	0.0225
3	0.8917	mean (no score imputation)	wk_10	Binomial	0.7	0.0289
4	0.9378	mean (no score imputation)	wk_10	Binomial	1.0	0.0203
5	0.9572	mean (no score imputation)	wk_10	Binomial	0.6	0.0338
6	1.0503	mean (no score imputation)	wk_10	Binomial	0.5	0.0405
7	1.1227	mean (no score imputation)	wk_10	Binomial	0.4	0.0506
8	1.1800	mean (no score imputation)	wk_9-10	Binomial	0.1	0.2025
9	1.1937	mean (no score imputation)	wk_10	Binomial	0.3	0.0675
10	1.2774	mean (no score imputation)	wk_10	Binomial	0.2	0.1013
11	1.2914	mean (no score imputation)	wk_10	Binomial	0.0	0.0203
12	1.3660	mean (no score imputation)	wk_10	Binomial	0.1	0.2025
13	1.4435	mean (no score imputation)	all	Binomial	0.1	0.2406
14	1.4712	mean (no score imputation)	wk_9-10	Binomial	0.2	0.1013
15	1.6436	mean (no score imputation)	wk_9-10	Binomial	0.3	0.0675

Table 21: PANAS Negative Regression Grid Search Results (only top 15 models shown)

Rank	CV Score (Total)	Imputer Strategy	Feature Subset	GLM Family	Alpha	Best Lambda	Theta	Tweedie Power
1	71.4550	mean (no score imputation)	all	Tweedie	0.6	2.1457	NA	1.3
2	72.8355	mean (no score imputation)	all	Tweedie	0.3	4.2914	NA	1.3
3	75.4575	mean (no score imputation)	all	Tweedie	0.6	0.9196	NA	1.7
4	77.2956	mean (no score imputation)	all	Tweedie	0.4	3.2185	NA	1.3
5	78.0083	mean (no score imputation)	all	Tweedie	0.1	12.8742	NA	1.3
6	79.8725	mean (no score imputation)	all	Tweedie	0.2	2.1457	NA	1.9
7	80.2377	mean (no score imputation)	all	Tweedie	0.9	1.4305	NA	1.3
8	80.4312	mean (no score imputation)	all	Tweedie	1.0	1.2874	NA	1.3
9	82.0877	mean (no score imputation)	all	Tweedie	0.7	1.8392	NA	1.3
10	82.4014	mean (no score imputation)	all	Tweedie	0.8	1.6093	NA	1.3
11	82.9919	mean (no score imputation)	all	Poisson	0.2	1.9311	NA	NA
12	84.5136	mean (no score imputation)	all	Poisson	0.1	3.8623	NA	NA
13	87.1342	mean (no score imputation)	all	Negative binomial	0.3	1.2874	1.00E-10	NA
14	87.1342	mean (no score imputation)	all	Negative binomial	0.3	1.2874	1.00E-08	NA
15	87.1417	mean (no score imputation)	all	Negative binomial	0.3	1.2846	0.0001	NA

7.4 Model 2: SVM

Table 22: Flourishing Scale Classification Grid Search Results (only top 15 models shown)

Rank	CV score (mean)	CV score (standard deviation)	PCA prop. var.	Imputer strategy	Feature subset	SVM kernel	SVM gamma	SVM degree	SVM C
1	0.510302	0.142735	0.95	median	wk_10	poly	0.01	4	1000
2	0.53754	0.13311	0.9	median	wk_10	linear	100	6	0.01
3	0.542169	0.159861	0.95	median	wk_10	linear	100	6	0.01
4	0.551876	0.121708	0.6	most_frequent	wk_10	linear	1	6	1
5	0.554259	0.126718	0.6	most_frequent	wk_10	linear	10	4	1
6	0.558497	0.162805	0.95	median	wk_10	linear	100	3	0.01
7	0.558808	0.126807	0.6	most_frequent	wk_10	linear	0.1	4	1
8	0.5639	0.149375	0.9	most_frequent	wk_9_10	linear	0.1	2	0.1
9	0.564098	0.155926	0.7	most_frequent	wk_9_10	linear	10	5	0.1
10	0.565822	0.139589	0.6	most_frequent	wk_10	linear	100	4	1
11	0.571048	0.114512	0.6	most_frequent	wk_10	linear	0.01	4	1
12	0.57111	0.154032	0.85	median	wk_10	poly	10	6	0.1
13	0.571479	0.090552	0.6	most_frequent	wk_10	linear	100	5	10
14	0.572356	0.170086	0.95	most_frequent	wk_10	linear	10	4	0.1
15	0.573069	0.100024	0.85	mean	wk_10	poly	0.01	6	1000

Table 23: Flourishing Scale Regression Grid Search Results (only top 15 models shown)

Rank	CV score (mean)[4]	CV score (standard deviation)	PCA prop. var.	Imputer strategy	Feature subset	SVM kernel	SVM gamma	SVM degree	SVM C
1	76.71637	74.41566	0.7	most_frequent	wk_9_10	linear	100	4	0.01
2	76.71637	74.41566	0.7	most_frequent	wk_9_10	linear	1	4	0.01
3	76.71637	74.41566	0.7	most_frequent	wk_9_10	linear	1	5	0.01
4	76.71637	74.41566	0.7	most_frequent	wk_9_10	linear	0.01	2	0.01
5	76.71637	74.41566	0.7	most_frequent	wk_9_10	linear	10	5	0.01
6	76.71637	74.41566	0.7	most_frequent	wk_9_10	linear	100	2	0.01
7	76.71637	74.41566	0.7	most_frequent	wk_9_10	linear	0.01	2	0.01
8	76.71637	74.41566	0.7	most_frequent	wk_9_10	linear	100	2	0.01
9	76.71637	74.41566	0.7	most_frequent	wk_9_10	linear	0.1	2	0.01
10	76.71637	74.41566	0.7	most_frequent	wk_9_10	linear	100	3	0.01
11	76.71637	74.41566	0.7	most_frequent	wk_9_10	linear	0.1	3	0.01
12	76.71637	74.41566	0.7	most_frequent	wk_9_10	linear	10	6	0.01
13	76.71637	74.41566	0.7	most_frequent	wk_9_10	linear	100	4	0.01
14	76.71637	74.41566	0.7	most_frequent	wk_9_10	linear	10	3	0.01
15	76.71637	74.41566	0.7	most_frequent	wk_9_10	linear	0.01	4	0.01

Table 24: PANAS Positive Classification Grid Search Results (only top 15 models shown)

[4] The top 15 scores are all the same as the top model uses a linear SVM kernel and a grid search is being conducted across the gamma and degree hyperparameters which are not used by the linear kernel. Whilst inefficient, given we performed 20,000 random grid searches across all the hyper-parameters it does not matter. Looking at the top models beyond 15 shows variation in the CV scores.

Rank	CV score (mean)	CV score (standard deviation)	Target imputation	PCA prop. var.	Imputer strategy	Feature subset	SVM kernel	SVM gamma	SVM degree	SVM C
1	0.565061	0.135897	No Imputation	0.95	most_frequent	wk_10	rbf	100	6	10
2	0.565086	0.125511	KNN Imputation	0.6	median	wk_10	poly	100	5	1
3	0.567691	0.191981	No Imputation	0.85	most_frequent	wk_10	poly	10	5	100
4	0.569316	0.131805	No Imputation	0.6	median	wk_10	rbf	100	2	1000
5	0.569353	0.144554	No Imputation	0.6	mean	wk_10	rbf	100	2	1000
6	0.569758	0.132904	No Imputation	0.95	median	wk_10	rbf	10	3	1
7	0.571927	0.133703	No Imputation	0.95	most_frequent	wk_10	rbf	10	5	1000
8	0.572807	0.135372	No Imputation	0.95	mean	wk_10	rbf	10	5	1
9	0.573675	0.133742	No Imputation	0.6	median	wk_10	rbf	100	6	100
10	0.574445	0.143146	No Imputation	0.95	median	wk_10	rbf	1	4	1
11	0.57465	0.132034	No Imputation	0.95	mean	wk_10	rbf	1	4	1
12	0.574765	0.12775	No Imputation	0.85	mean	wk_10	rbf	1	3	1
13	0.574866	0.128089	No Imputation	0.9	most_frequent	wk_10	rbf	10	2	10
14	0.575364	0.127556	No Imputation	0.7	most_frequent	wk_10	rbf	10	4	1000
15	0.575953	0.131955	No Imputation	0.8	median	wk_10	rbf	100	5	10

Table 25: PANAS Positive Regression Grid Search Results (only top 15 models shown)

Rank	CV score (mean)	CV score (standard deviation)	Target imputation	PCA prop. var.	Imputer strategy	Feature subset	SVM kernel	SVM gamma	SVM degree	SVM C
1	41.75514	46.26143	KNN Imputation	0.6	median	wk_9_10	rbf	0.01	4	10
2	41.75514	46.26143	KNN Imputation	0.6	median	wk_9_10	rbf	0.01	2	10
3	41.75514	46.26143	KNN Imputation	0.6	median	wk_9_10	rbf	0.01	6	10
4	42.02818	26.75861	No Imputation	0.7	median	wk_10	rbf	0.1	3	10
5	42.02818	26.75861	No Imputation	0.7	median	wk_10	rbf	0.1	4	10
6	42.02818	26.75861	No Imputation	0.7	median	wk_10	rbf	0.1	6	10
7	42.02838	26.8603	No Imputation	0.7	median	wk_9_10	rbf	0.1	2	1000
8	42.02838	26.8603	No Imputation	0.7	median	wk_9_10	rbf	0.1	4	1000
9	42.02838	26.8603	No Imputation	0.7	median	wk_9_10	rbf	0.1	6	100
10	42.02838	26.8603	No Imputation	0.7	median	wk_9_10	rbf	0.1	2	100
11	42.02838	26.8603	No Imputation	0.7	median	wk_9_10	rbf	0.1	3	1000
12	42.09859	42.52865	KNN Imputation	0.7	median	wk_9_10	rbf	0.01	4	10
13	42.09859	42.52865	KNN Imputation	0.7	median	wk_9_10	rbf	0.01	2	10
14	42.09859	42.52865	KNN Imputation	0.7	median	wk_9_10	rbf	0.01	6	10
15	42.09859	42.52865	KNN Imputation	0.7	median	wk_9_10	rbf	0.01	3	10

Table 26: PANAS Negative Classification Grid Search Results (only top 15 models shown)

Rank	CV score (mean)	CV score (standard deviation)	PCA prop. var.	Imputer strategy	Feature subset	SVM kernel	SVM gamma	SVM degree	SVM C
1	0.458781	0.211216	0.7	mean	all	rbf	0.01	3	1
2	0.475114	0.229672	0.6	mean	all	rbf	0.01	6	1
3	0.476959	0.222539	0.6	median	all	rbf	0.01	4	1
4	0.494119	0.195778	0.7	mean	all	rbf	0.01	2	1

5	0.498949	0.191469	0.6	median	all	rbf	0.01	5	1
6	0.500249	0.187092	0.6	mean	all	rbf	0.01	5	1
7	0.503514	0.194776	0.7	mean	all	rbf	0.01	5	1
8	0.504652	0.164466	0.7	mean	all	rbf	0.01	5	1
9	0.504862	0.195267	0.6	mean	all	rbf	0.01	4	1
10	0.51455	0.153436	0.7	median	all	rbf	0.01	2	1
11	0.524123	0.184121	0.6	median	all	rbf	0.01	5	1
12	0.526804	0.171823	0.6	median	all	rbf	0.01	2	1
13	0.530199	0.149432	0.6	mean	all	rbf	0.01	4	1
14	0.531489	0.209684	0.8	mean	all	rbf	0.01	5	1
15	0.532331	0.214252	0.95	median	wk_10	poly	1	4	1

Table 27: PANAS Negative Regression Grid Search Results (only top 15 models shown)

Rank	**CV score (mean)**	**CV score (standard deviation)**	**PCA prop. var.**	**Imputer strategy**	**Feature subset**	**SVM kernel**	**SVM gamma**	**SVM degree**	**SVM C**
1	44.97851	25.25375	0.6	median	wk_10	rbf	0.01	2	10
2	44.97851	25.25375	0.6	median	wk_10	rbf	0.01	3	10
3	44.97851	25.25375	0.6	median	wk_10	rbf	0.01	5	10
4	44.97851	25.25375	0.6	median	wk_10	rbf	0.01	5	10
5	44.97851	25.25375	0.6	median	wk_10	rbf	0.01	6	10
6	44.97851	25.25375	0.6	median	wk_10	rbf	0.01	6	10
7	47.7929	23.77891	0.6	mean	wk_10	rbf	0.01	6	100
8	47.7929	23.77891	0.6	mean	wk_10	rbf	0.01	5	100
9	47.7929	23.77891	0.6	mean	wk_10	rbf	0.01	3	100
10	47.7929	23.77891	0.6	mean	wk_10	rbf	0.01	4	100
11	48.77909	26.64412	0.6	mean	wk_10	rbf	0.01	2	10
12	48.77909	26.64412	0.6	mean	wk_10	rbf	0.01	4	10
13	48.77909	26.64412	0.6	mean	wk_10	rbf	0.01	6	10
14	48.77909	26.64412	0.6	mean	wk_10	rbf	0.01	4	10
15	48.77909	26.64412	0.6	mean	wk_10	rbf	0.01	5	10

7.5 Model 3: Gradient Boosting

Table 21: Flourishing Scale Classification Grid Search Results (only top 15 models shown)

Rank	**CV score (mean)**	**CV score (standard deviation)**	**Target imputation**	**Learning rate**	**Gamma**	**Max depth**	**Max features in each tree (split)**	**Subsample**	**Min child weight**	**Number of estimators (boost rounds)**	**Alpha**	**Lambda**
1	0.497496	0.256252	KNN Imputation	0.1	0	1	0.3	1	2	100	0	1
2	0.497496	0.256252	No Imputation	0.1	0	1	0.3	1	2	100	0	1
3	0.506982	0.408991	No Imputation	0.1	0.01	7	0.3	1	1	1000	0	1
4	0.506982	0.408991	KNN Imputation	0.1	0.01	7	0.3	1	1	1000	0	1
5	0.510359	0.257673	No Imputation	0.01	0.1	7	0.7	1	2	500	0	1
6	0.510359	0.257673	KNN Imputation	0.01	0.1	7	0.7	1	2	500	0	1
7	0.516306	0.40169	KNN Imputation	0.1	0.01	3	1	1	1	1000	0	1
8	0.516306	0.40169	No Imputation	0.1	0.01	3	1	1	1	1000	0	1

9	0.522289	0.4173	No Imputation	0.01	0.05	3	0.5	1	1	1000	0	1
10	0.522289	0.4173	KNN Imputation	0.01	0.05	3	0.5	1	1	1000	0	1
11	0.528602	0.235711	No Imputation	0.1	0.01	3	0.5	0.7	2	1500	0	1
12	0.528602	0.235711	KNN Imputation	0.1	0.01	3	0.5	0.7	2	1500	0	1
13	0.533232	0.201317	KNN Imputation	0.01	0	1	0.3	0.7	2	1500	0	1
14	0.533232	0.201317	No Imputation	0.01	0	1	0.3	0.7	2	1500	0	1
15	0.540754	0.245483	No Imputation	0.1	0.1	7	0.7	0.7	2	500	0	1

Table 22: Flourishing Scale Regression Grid Search Results (only top 15 models shown)

Rank	CV score (mean)	CV score (standard deviation)	Target imputation	Learning rate	Gamma	Max depth	Max features in each tree (split)	Subsample	Min child weight	Number of estimators (boost rounds)	Alpha	Lambda
1	89.42315	60.35462	No Imputation	0.01	0.05	1	0.3	0.3	1	300	1.00E-05	0.01
2	89.42315	60.35462	KNN Imputation	0.01	0.05	1	0.3	0.3	1	300	1.00E-05	0.01
3	99.49536	81.59688	No Imputation	0.01	0.1	1	0.5	0.3	1	1000	1.00E-05	0.45
4	99.49536	81.59688	KNN Imputation	0.01	0.1	1	0.5	0.3	1	1000	1.00E-05	0.45
5	100.23814	62.45249	KNN Imputation	0.01	0.01	1	0.7	0.3	2	300	0.01	0.45
6	100.23814	62.45249	No Imputation	0.01	0.01	1	0.7	0.3	2	300	0.01	0.45
7	105.45715	77.54336	No Imputation	0.1	0.1	7	0.7	0.3	3	300	0.75	0.01
8	105.45715	77.54336	KNN Imputation	0.1	0.1	7	0.7	0.3	3	300	0.75	0.01
9	107.17013	84.08640	No Imputation	0.01	0.01	3	0.3	0.5	2	500	1.00E-05	0.45
10	107.17013	84.08640	KNN Imputation	0.01	0.01	3	0.3	0.5	2	500	1.00E-05	0.45
11	107.35256	83.18527	No Imputation	0.1	0.01	5	0.7	0.3	1	1500	1.00E-05	0.45
12	107.35256	83.18527	KNN Imputation	0.1	0.01	5	0.7	0.3	1	1500	1.00E-05	0.45
13	108.86356	85.90307	No Imputation	0.01	0.01	5	0.7	0.3	2	1000	0.75	1.00E-05
14	108.86356	85.90307	KNN Imputation	0.01	0.01	5	0.7	0.3	2	1000	0.75	1.00E-05
15	112.4706	99.34004	KNN Imputation	0.01	0.01	3	0.3	1	2	1500	1.00E-05	1.00E-05

Table 23: PANAS Positive Classification Grid Search Results (only top 15 models shown)

Rank	CV score (mean)	CV score (standard deviation)	Target imputation	Learning rate	Gamma	Max depth	Max features in each tree (split)	Subsample	Min child weight	Number of estimators (boost rounds)	Alpha	Lambda
1	0.65388	0.11639	KNN Imputation	0.001	0.1	7	0.7	1	2	500	0	1
2	0.66259	0.10830	KNN Imputation	0.001	0.01	3	0.5	1	2	500	0	1
3	0.66332	0.11757	KNN Imputation	0.001	0	1	1	1	1	500	0	1
4	0.66655	0.07600	KNN Imputation	0.001	0	1	0.7	1	2	300	0	1
5	0.66970	0.38998	KNN Imputation	0.1	0.01	3	0.5	0.7	2	1500	0	1
6	0.67603	0.42059	KNN Imputation	0.1	0.1	7	0.7	0.7	2	500	0	1
7	0.67750	0.27606	KNN Imputation	0.01	0	5	0.5	0.7	2	500	0	1
8	0.67816	0.33288	KNN Imputation	0.01	0	1	0.3	0.7	2	1500	0	1
9	0.67829	0.09142	KNN Imputation	0.001	0.1	3	0.3	1	2	500	0	1
10	0.68129	0.12092	KNN Imputation	0.001	0.01	1	0.3	0.7	2	1500	0	1
11	0.68180	0.07825	KNN Imputation	0.001	0	1	1	0.7	1	500	0	1
12	0.68199	0.09572	KNN Imputation	0.001	0.1	5	0.5	0.7	2	1000	0	1

13	0.68229	0.07683	KNN Imputation	0.001	0.05	7	0.5	0.7	1	500	0	1
14	0.68262	0.07001	KNN Imputation	0.001	0.01	5	1	1	2	300	0	1
15	0.68272	0.05427	KNN Imputation	0.001	0	1	0.7	0.7	2	500	0	1

Table 24: PANAS Positive Regression Grid Search Results (only top 15 models shown)

Rank	CV score (mean)	CV score (standard deviation)	Target imputation	Learning rate	Gamma	Max depth	Max features in each tree (split)	Subsample	Min child weight	Number of estimators (boost rounds)	Alpha	Lambda
1	36.29245	19.13942	KNN Imputation	0.01	0.1	1	0.5	0.7	3	1500	0.75	1.00E-05
2	36.29245	19.13942	No Imputation	0.01	0.1	1	0.5	0.7	3	1500	0.75	1.00E-05
3	36.82154	20.41038	KNN Imputation	0.01	0.05	1	0.5	0.7	2	1500	0.01	1.00E-05
4	36.82154	20.41038	No Imputation	0.01	0.05	1	0.5	0.7	2	1500	0.01	1.00E-05
5	36.95040	19.36132	No Imputation	0.01	0.05	1	0.7	0.5	1	1500	0.75	0.45
6	36.95040	19.36132	KNN Imputation	0.01	0.05	1	0.7	0.5	1	1500	0.75	0.45
7	37.53505	23.81167	KNN Imputation	0.1	0.1	7	1	0.3	2	1500	0.01	0.45
8	37.53505	23.81167	No Imputation	0.1	0.1	7	1	0.3	2	1500	0.01	0.45
9	37.56048	22.16129	KNN Imputation	0.1	0	5	0.3	0.5	3	300	1.00E-05	0.45
10	37.56048	22.16129	No Imputation	0.1	0	5	0.3	0.5	3	300	1.00E-05	0.45
11	39.44521	15.68037	No Imputation	0.1	0.01	3	0.3	0.5	2	1000	1.00E-05	0.45
12	39.44521	15.68037	KNN Imputation	0.1	0.01	3	0.3	0.5	2	1000	1.00E-05	0.45
13	39.89918	25.54190	KNN Imputation	0.1	0.05	1	0.7	0.3	2	500	1.00E-05	0.45
14	39.89918	25.54190	No Imputation	0.1	0.05	1	0.7	0.3	2	500	1.00E-05	0.45
15	40.66421	20.61046	No Imputation	0.01	0.1	7	0.3	0.5	2	1500	0.75	0.45

Table 25: PANAS Negative Classification Grid Search Results (only top 15 models shown)

Rank	CV score (mean)	CV score (standard deviation)	Target imputation	Learning rate	Gamma	Max depth	Max features in each tree (split)	Subsample	Min child weight	Number of estimators (boost rounds)	Alpha	Lambda
1	0.59951	0.16173	No Imputation	0.001	0.05	7	0.5	1	1	1000	0	1
2	0.59951	0.16173	KNN Imputation	0.001	0.05	7	0.5	1	1	1000	0	1
3	0.64191	0.12850	KNN Imputation	0.001	0.05	3	0.7	0.7	1	1500	0	1
4	0.64191	0.12850	No Imputation	0.001	0.05	3	0.7	0.7	1	1500	0	1
5	0.64221	0.08027	KNN Imputation	0.001	0	1	1	1	1	500	0	1
6	0.64221	0.08027	No Imputation	0.001	0	1	1	1	1	500	0	1
7	0.64290	0.06437	KNN Imputation	0.001	0.1	7	0.7	1	2	500	0	1
8	0.64290	0.06437	No Imputation	0.001	0.1	7	0.7	1	2	500	0	1
9	0.64341	0.05521	KNN Imputation	0.001	0.01	3	0.5	1	2	500	0	1
10	0.64341	0.05521	No Imputation	0.001	0.01	3	0.5	1	2	500	0	1
11	0.65166	0.05105	KNN Imputation	0.001	0.1	3	0.3	1	2	500	0	1
12	0.65166	0.05105	No Imputation	0.001	0.1	3	0.3	1	2	500	0	1
13	0.65297	0.04179	No Imputation	0.001	0	1	0.7	1	2	300	0	1
14	0.65297	0.04179	KNN Imputation	0.001	0	1	0.7	1	2	300	0	1
15	0.65923	0.04972	KNN Imputation	0.001	0.05	7	0.5	0.7	1	500	0	1

Table 26: PANAS Negative Regression Grid Search Results (only top 15 models shown)

Rank	CV score (mean)	CV score (standard deviation)	Target imputation	Learning rate	Gamma	Max depth	Max features in each tree (split)	Subsample	Min child weight	Number of estimators (boost rounds)	Alpha	Lambda
1	39.66720	14.47942	KNN Imputation	0.01	0.01	5	0.3	0.7	1	1000	0.01	0.01
2	39.66720	14.47942	No Imputation	0.01	0.01	5	0.3	0.7	1	1000	0.01	0.01
3	42.82522	18.17954	KNN Imputation	0.01	0.01	3	0.3	1	2	1500	1.00E-05	1.00E-05
4	42.82522	18.17954	No Imputation	0.01	0.01	3	0.3	1	2	1500	1.00E-05	1.00E-05
5	44.45177	17.80788	No Imputation	0.01	0.01	7	0.5	0.7	2	1000	0.01	1.00E-05
6	44.45177	17.80788	KNN Imputation	0.01	0.01	7	0.5	0.7	2	1000	0.01	1.00E-05
7	44.54814	20.71448	KNN Imputation	0.01	0	1	0.3	1	2	300	1.00E-05	0.01
8	44.54814	20.71448	No Imputation	0.01	0	1	0.3	1	2	300	1.00E-05	0.01
9	47.38244	26.82938	No Imputation	0.01	0	7	0.3	0.7	2	300	0.75	0.45
10	47.38244	26.82938	KNN Imputation	0.01	0	7	0.3	0.7	2	300	0.75	0.45
11	47.98194	19.74136	KNN Imputation	0.1	0.05	3	0.7	1	3	100	0.75	0.01
12	47.98194	19.74136	No Imputation	0.1	0.05	3	0.7	1	3	100	0.75	0.01
13	48.73336	18.88523	No Imputation	0.1	0	5	0.3	0.5	3	300	1.00E-05	0.45
14	48.73336	18.88523	KNN Imputation	0.1	0	5	0.3	0.5	3	300	1.00E-05	0.45
15	49.11841	17.94749	KNN Imputation	0.1	0	1	0.3	0.7	2	100	1.00E-05	0.45

7.6 Model 4: KNN

Table 28 - KNN Flourishing Scale Classification Grid Search Results

Rank	CV Score (mean)	CV Score (standard deviation)	PCA prop. var.	Imputer strategy	Feature subset	KNN n neighbours	KNN weights	KNN metric
1	0.60607	0.241896	0.95	most_frequent	wk_10	15	uniform	euclidean
2	0.60656	0.234023	0.8	most_frequent	wk_10	12	uniform	manhattan
3	0.60707	0.31409	0.99	most_frequent	wk_10	13	uniform	euclidean
4	0.60966	0.165002	0.7	most_frequent	wk_10	15	uniform	euclidean
5	0.60967	0.170619	0.95	most_frequent	wk_10	16	uniform	manhattan
6	0.61048	0.15831	0.7	most_frequent	wk_10	17	uniform	manhattan
7	0.61357	0.125996	0.95	most_frequent	wk_10	17	uniform	manhattan
8	0.61361	0.120196	0.7	most_frequent	wk_10	19	uniform	euclidean
9	0.61472	0.161116	0.85	most_frequent	wk_9_10	17	uniform	euclidean
10	0.61764	0.253683	0.85	most_frequent	wk_10	14	uniform	euclidean
11	0.62093	0.175071	0.95	most_frequent	wk_9_10	16	uniform	euclidean
12	0.62143	0.258634	0.85	most_frequent	wk_10	11	uniform	euclidean
13	0.62389	0.227952	0.8	most_frequent	wk_10	13	uniform	euclidean
14	0.62529	0.134092	0.99	most_frequent	wk_10	17	uniform	manhattan
15	0.62615	0.146814	0.6	most_frequent	wk_10	18	uniform	euclidean

Table 29 - KNN Flourishing Scale Regression Grid Search Results

Rank	CV Score (mean)	CV Score (standard deviation)	PCA prop. var.	Imputer strategy	Feature subset	KNN n neighbours	KNN weights	KNN metric
1	67.9234	64.1468	0.85	most_frequent	wk_10	11	uniform	euclidean
2	68.2972	65.35152	0.6	most_frequent	wk_9_10	17	uniform	euclidean
3	68.4494	50.59189	0.99	most_frequent	all	5	distance	euclidean

4	69.2173	57.42126	0.99	most_frequent	wk_10	7	uniform	manhattan
5	69.4561	58.94374	0.95	most_frequent	wk_10	12	uniform	manhattan
6	69.6181	65.1848	0.8	most_frequent	wk_10	13	uniform	euclidean
7	70.0872	49.1873	0.7	most_frequent	wk_10	8	uniform	euclidean
8	70.1213	71.75828	0.9	median	all	4	distance	manhattan
9	70.1239	72.42933	0.95	most_frequent	all	4	distance	euclidean
10	70.351	68.27362	0.7	most_frequent	wk_10	15	uniform	euclidean
11	70.3757	58.35865	0.8	most_frequent	wk_9_10	14	uniform	euclidean
12	70.5966	69.13102	0.99	mean	all	6	distance	manhattan
13	70.6307	55.59043	0.99	median	all	5	uniform	manhattan
14	70.8304	63.49228	0.99	most_frequent	wk_10	13	uniform	euclidean
15	70.8477	61.35311	0.8	most_frequent	wk_10	10	uniform	manhattan

Table 30 - KNN PANAS Negative Classification Grid Search Results

Rank	CV Score (mean)	CV Score (standard deviation)	PCA prop. var.	Imputer strategy	Feature subset	KNN n neighbours	KNN weights	KNN metric
1	0.59179	0.22461	0.6	most_frequent	wk_10	8	uniform	euclidean
2	0.59776	0.101165	0.7	median	wk_10	12	uniform	manhattan
3	0.6043	0.146477	0.8	most_frequent	wk_10	11	uniform	manhattan
4	0.60889	0.187031	0.7	most_frequent	wk_10	8	uniform	euclidean
5	0.61348	0.097119	0.8	median	wk_10	12	uniform	manhattan
6	0.6172	0.12432	0.99	mean	wk_9_10	7	distance	manhattan
7	0.61847	0.303192	0.6	most_frequent	wk_9_10	4	distance	manhattan
8	0.61912	0.214091	0.6	median	wk_9_10	6	distance	manhattan
9	0.62006	0.084489	0.6	median	wk_10	14	uniform	euclidean
10	0.62077	0.086043	0.6	median	wk_10	16	uniform	manhattan
11	0.62086	0.120481	0.8	most_frequent	wk_10	12	uniform	manhattan
12	0.62369	0.183444	0.6	median	wk_10	8	uniform	euclidean
13	0.6264	0.195195	0.8	most_frequent	wk_10	10	uniform	manhattan
14	0.63031	0.138815	0.6	mean	wk_10	10	uniform	manhattan
15	0.63035	0.081472	0.7	mean	wk_9_10	8	uniform	manhattan

Table 31 - KNN PANAS Negative Regression Grid Search Results

Rank	CV Score (mean)	CV Score (standard deviation)	PCA prop. var.	Imputer strategy	Feature subset	KNN n neighbours	KNN weights	KNN metric
1	46.8587	16.48116	0.99	median	wk_10	5	uniform	euclidean
2	47.17	16.26891	0.99	median	wk_10	4	distance	euclidean
3	49.3222	21.45633	0.85	median	wk_10	3	uniform	manhattan
4	49.6646	22.39693	0.99	mean	wk_10	4	uniform	manhattan
5	50.0178	25.11925	0.99	mean	wk_10	6	distance	manhattan
6	50.1667	25.15917	0.99	median	wk_10	3	uniform	euclidean
7	50.1963	26.53181	0.8	mean	wk_10	3	uniform	manhattan
8	50.845	26.3321	0.8	mean	wk_10	3	distance	manhattan
9	51.0024	20.45065	0.9	median	wk_10	8	distance	manhattan
10	51.0833	21.42098	0.9	median	wk_10	4	uniform	euclidean
11	51.1614	20.34162	0.99	mean	wk_10	10	distance	euclidean
12	51.2052	21.89335	0.9	median	wk_10	8	uniform	manhattan

13	51.2691	20.52274	0.99	mean	wk_10	10	distance	manhattan
14	51.4907	38.95363	0.6	most_frequent	wk_10	5	uniform	euclidean
15	51.5403	23.57339	0.8	mean	wk_10	11	distance	manhattan

Table 32 - KNN PANAS Positive Classification Grid Search Results

Rank	CV Score (mean)	CV Score (standard deviation)	PCA prop. var.	Imputer strategy	Feature subset	KNN n neighbours	KNN weights	KNN metric
1	0.628540145	0.146972702	0.6	median	wk_10	16	uniform	manhattan
2	0.628714281	0.072809157	0.6	median	wk_10	17	uniform	euclidean
3	0.631089861	0.113077488	0.6	mean	wk_10	14	uniform	manhattan
4	0.635098343	0.146127077	0.95	most_frequent	wk_10	16	uniform	manhattan
5	0.637337859	0.098628503	0.6	median	wk_10	16	uniform	euclidean
6	0.63734102	0.136530957	0.6	mean	wk_10	13	uniform	manhattan
7	0.638957972	0.154721567	0.6	median	wk_10	14	uniform	euclidean
8	0.639590291	0.179218483	0.7	median	wk_9_10	10	uniform	euclidean
9	0.642843998	0.09527556	0.6	most_frequent	all	16	uniform	manhattan
10	0.643344875	0.140827076	0.6	median	wk_10	15	uniform	euclidean
11	0.644509971	0.082047234	0.8	median	wk_10	18	uniform	manhattan
12	0.64524814	0.116654984	0.7	median	wk_10	12	uniform	manhattan
13	0.646815604	0.114644418	0.8	median	wk_10	15	uniform	euclidean
14	0.647408648	0.061176399	0.7	most_frequent	all	17	uniform	euclidean
15	0.649081547	0.067117149	0.99	mean	all	19	uniform	euclidean

Table 33 - KNN PANAS Positive Regression Grid Search Results

Rank	CV Score (mean)	CV Score (standard deviation)	PCA prop. var.	Imputer strategy	Feature subset	KNN n neighbours	KNN weights	KNN metric
1	36.74133333	22.31011187	0.99	mean	wk_10	5	uniform	manhattan
2	38.56533146	38.60057359	0.6	median	wk_9_10	5	distance	manhattan
3	39.28212957	40.09890484	0.6	median	wk_9_10	6	distance	manhattan
4	39.856	27.69000449	0.6	mean	wk_10	5	uniform	euclidean
5	40.2002152	43.17418419	0.6	median	wk_9_10	8	distance	euclidean
6	40.5125	25.63262793	0.99	mean	wk_10	4	uniform	manhattan
7	40.69195621	39.04524042	0.7	mean	wk_9_10	5	distance	manhattan
8	41.66266667	24.54989589	0.6	mean	wk_10	5	uniform	manhattan
9	41.82760417	27.25144763	0.7	most_frequent	wk_10	8	uniform	euclidean
10	41.91088435	34.28698767	0.9	median	wk_9_10	7	uniform	manhattan
11	41.97770189	37.75816133	0.7	mean	wk_9_10	6	distance	manhattan
12	42.01041667	24.75475653	0.6	most_frequent	wk_10	8	uniform	euclidean
13	42.01421937	39.84236591	0.6	mean	wk_9_10	7	distance	euclidean
14	42.04965986	30.59598117	0.8	most_frequent	wk_10	7	uniform	euclidean
15	42.11145833	36.23932525	0.7	mean	wk_9_10	8	uniform	manhattan

7.7 Results: Flourishing Score Classification (GLM)

Figure 10: Predicted vs. Observed by quintiles for training (left) and test (right) data

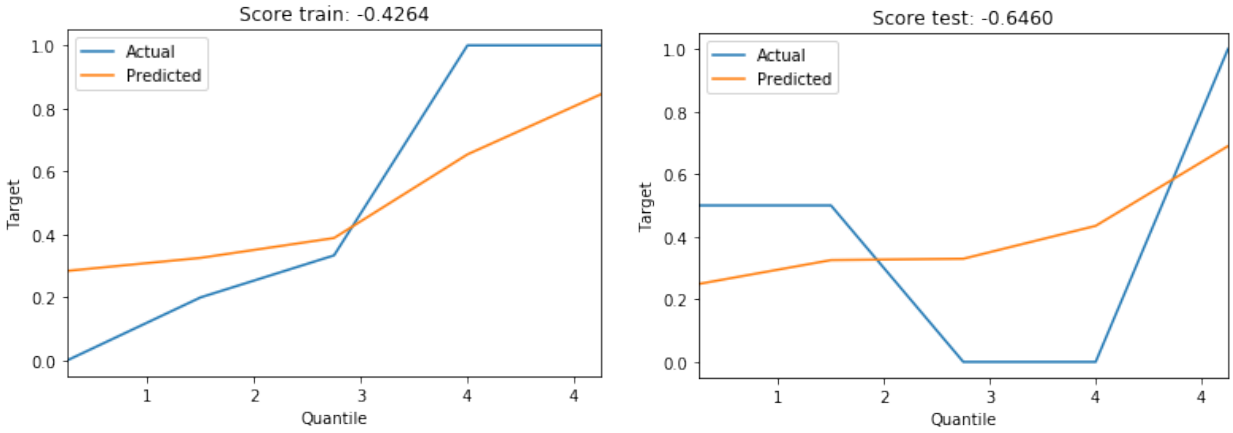
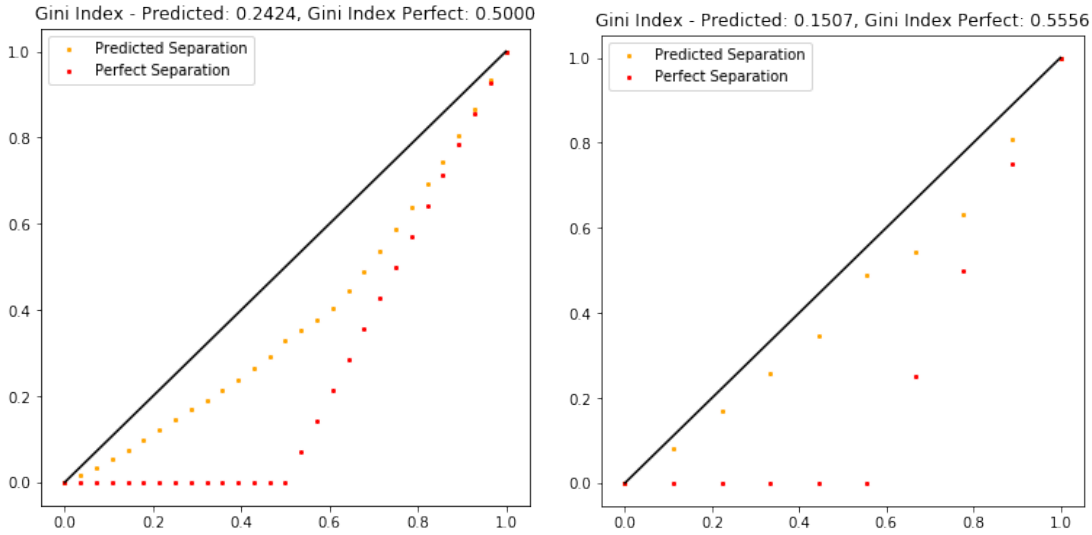


Figure 11: Lorenz Curves for training (left) and test (right) data



7.8 Results: PANAS Positive Classification (XGBoost)

Figure 12: Lorenz Curves for training (left) and test (right) data

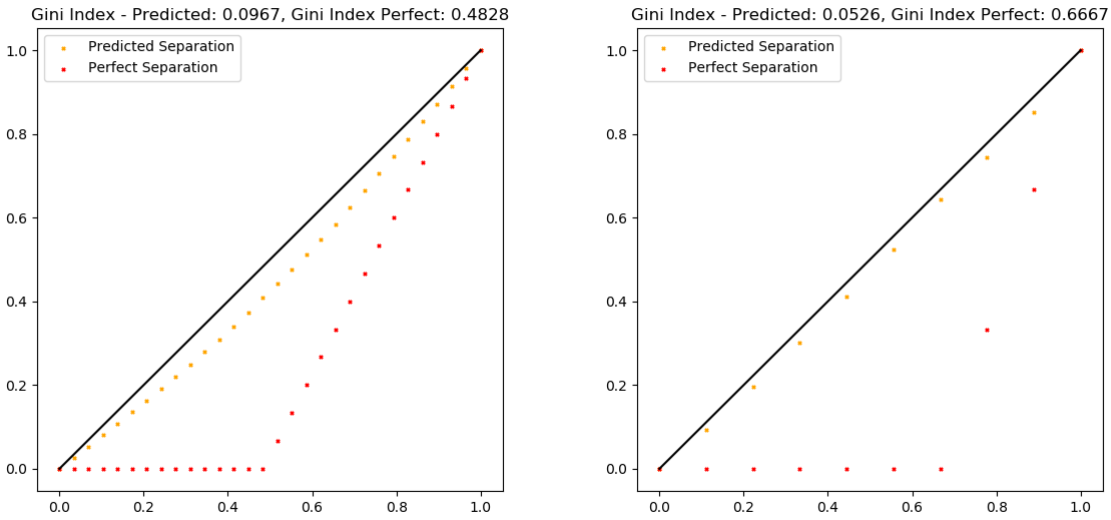
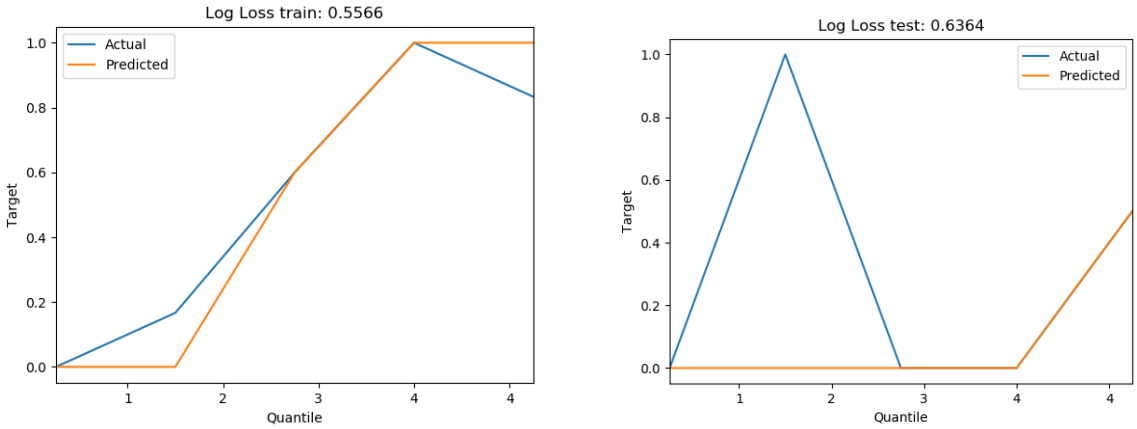


Figure 13: Predicted vs. Observed by quintiles for training (left) and test (right) data



7.9 Results: PANAS Negative Regression (SVM)

Figure 14: Predicted vs. Observed by quintiles for training (left) and test (right) data

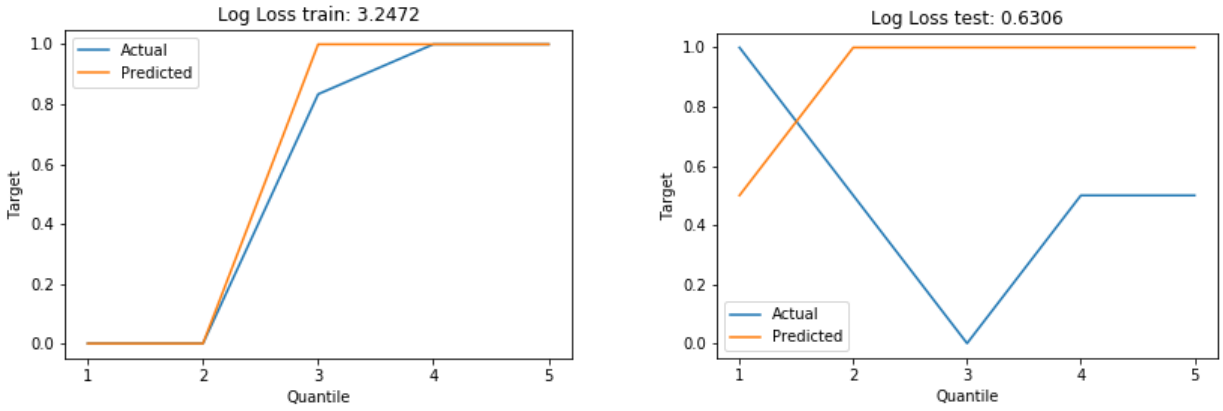


Figure 15: Lorenz Curves for training (left) and test (right) data

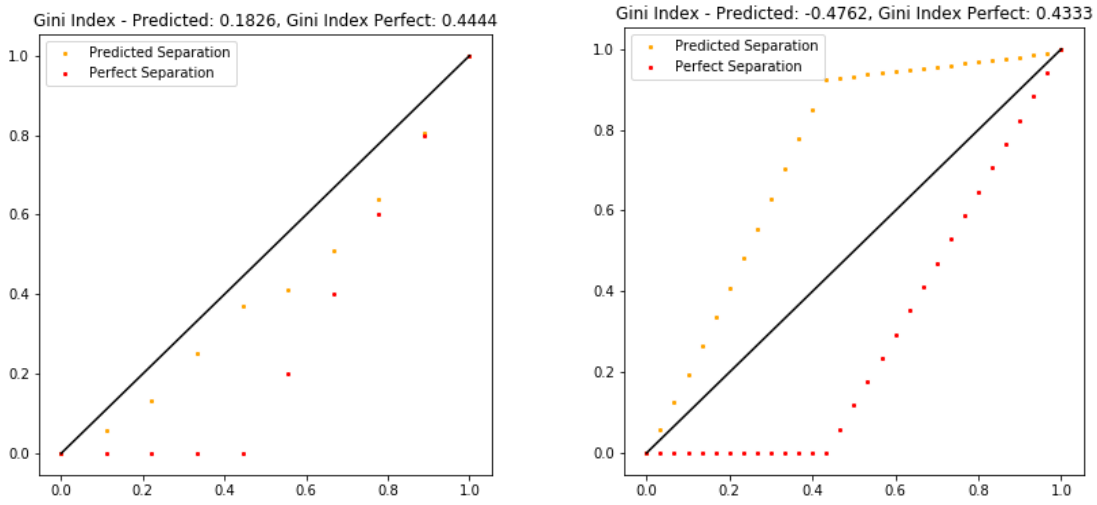
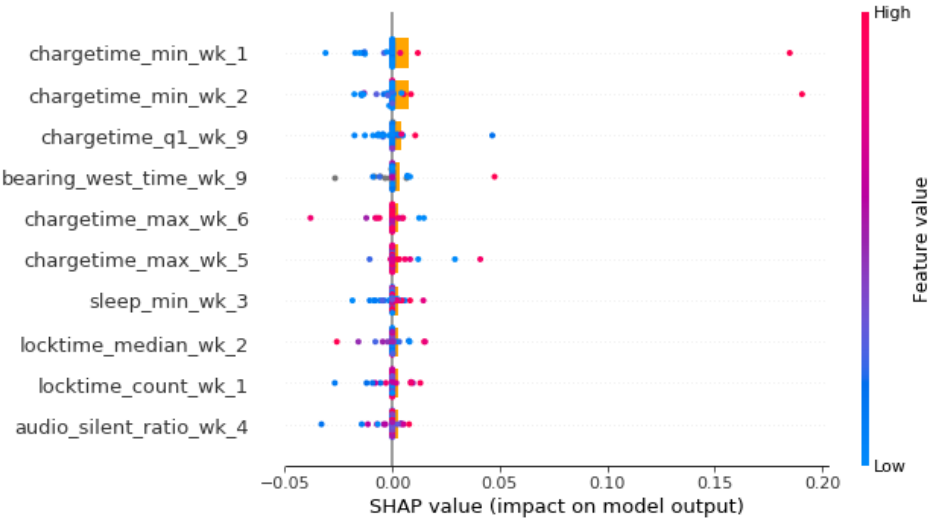


Figure 16: SVM SHAP values



9 References

Bergstra, J., & Bengio, Y. (2012). Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research*.

Blagus, R., & Lusa, L. (2013). SMOTE for high-dimensional class-imbalanced data. *BMC BioInformatics*.

Chen, Z., Lin, M., Chen, F., Lane, N. D., Cardone, G., Wang, R., . . . Campbell, A. T. (2013). Unobtrusive Sleep Monitoring using Smartphone. *7th International Conference on Pervasive Computing Technologies for Healthcare and Workshops*, 145-152.

DeVries, P. M., Viegas, F., Wattenberg, M., & Meade, B. J. (2018). Deep learning of aftershock patterns following large earthquakes. *Nature*, 632–634.

Dietrich, W. (1994). *A study of distance-based machine learning algorithms.* Retrieved from Oregon State University: https://ir.library.oregonstate.edu/concern/graduate_thesis_or_dissertations/zw12z7835

Friedman, J., Hastie, T., & Tibshirani, R. (2010, January). Regularization Paths for Generalized Linear Models via Coordinate Descent. *Journal of Statistical Software, 33*(1).

Geron, A. (2017). *Hands-on Machine Learning with Sciki-Learn and TensorFlow.* Sebastopol: O'Reilly.

Guestrin, C., & Chen, T. (2016). XGBoost: A Scalable Tree Boosting System. *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785-794). San Francisco: kdd.

H2O. (2019, November). *H2O Interface for Python, 3.18.0.2*.

Hastie, T., Tibshirani, R., & Friedman, J. (2016). The Elements of Statistical Learning. Springer.

James, G., Witten, D., Hastie, T., & Tibshiranie, R. (2013). *An Introduction to Statistical Learning.* New York: Springer.

Kouiroukidis, N., & Evangelidis, G. (2011). The effects of dimensionality curse in high dimensional knn search. *2011 15th Panhellenic Conference on Informatics* (pp. 41-45). Kastonia, Greece: IEEE.

Lundberg, S., & Lee, S.-I. (2017). A Unified Approach to Interpreting Model Predictions. *Conference on Neural Information Processing Systems*.

Mignan, A., & Broccardo, M. (2019). One neuron versus deep learning in aftershock prediction. *Nature*, E1–E3.

Muller, A., & Guide, S. (2016). *Introduction to Machine Learning with Python.* Sebastopol: O'Reilly.

Narayana, A. (2019). *How to recognize AI snake oil.* Retrieved from https://www.cs.princeton.edu/~arvindn/talks/MIT-STS-AI-snakeoil.pdf

scikit-learn SVM documentation. (2019, November 24). Retrieved from scikit-learn: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

Sharma, A., Madaan, V., & Petty, F. D. (2006). Exercise for mental health. *Primary care companion to the Journal of clinical psychiatry, 8*(2), 106. doi:10.4088/pcc.v08n0208a

Wang, R., Chen, F., Chen, Z., Li, T., Harari, G., Tignor, S., . . . Campbell, A. T. (2014). StudentLife: assessing mental health, academic performance and behavioral trends of college students using smartphones. *Proceedings of the 2014 ACM international joint conference on pervasive and ubiquitous computing* (pp. 3-14). Seattle, Washington: ACM.

XGBoost Documentation. (2019, November 25). Retrieved from XGBoost: https://xgboost.readthedocs.io/en/latest/index.html