Projectreport Machine Learning 2

Maluna Menke, Ari (Sara) Wahl, Pavlo Kravets

2024-01-01

Contents

1. Introduction

Our general idea was to work with LGBT-related data. This was not as easy as expected, since it seems there are not a lot of datasets openly available that have that kind of information. Finally, we found a US survey by the CDC, that regularly monitors the country's youth in a lot of dimensions, but among other questions also asks for sexual experiences and identification.

2. The Dataset

"The *Youth Risk Behavior Survey (YRBS)* measures health-related behaviors and experiences that can lead to death and disability among youth and adults.[…] Some of the health-related behaviors and experiences monitored are: - Student demographics: sex, sexual identity, race and ethnicity, and grade - Youth health behaviors and conditions: sexual, injury and violence, bullying, diet and physical activity, obesity, and mental health, including suicide - Substance use behaviors: electronic vapor product and tobacco product use, alcohol use, and other drug use - Student experiences: parental monitoring, school connectedness, unstable housing, and exposure to community violence [1]. It is a national survey conducted by the CDC (Center for Disease Control and Prevention) and includes high school students from both private and public schools within the U.S. Data is collected from 1991 through 2021, we are only using the most recent data from 2021. If you want to learn more about the data there is an accompanying Data User Guide.[2].

2.1 Preprocessing of the dataset

To preprocess the dataset, we first ran a summary of our dataset. The number of NAs seems to depend very much on the question. The variable "orig_rec" only contained NAs and has therefore been removed, as well as the variable "site" which only contained "XX" entries. Variables q4 and q5 are already aggregated in "raceeth" and have also been deleted. The variable "record" seems to be an ID for the observations. This has to be considered later.

2.2 Missing Data

We will first exclude all the observations with NAs in all the target-related variables q25 to q29. Since we want to build our target variable on these questions, the target variable cannot all be empty. The amount of data available should be enough to just exclude these observations. After removing the observations that have NAs in all the variables, that are used to create our target variable, we still have around 13.7% NAs in the dataset.

What if we had just excluded every NA in the dataset? We will try and see if this is a viable option, since this woul not just be quick and easy, but we would also just have "real" answers. The exclusion of NAs leads to a severe reduction in the number of observations. The original data consisted of 17232 observations, after reducing the target-related NAs only, we have 11753 observations left. If we omit all NAs, the reduced dataframe still has 4334 observations.

In this case need to assess the loss of information foremost about our target variable. The important question is if there is a pattern to the missingness in our data, not just, but especially about our target variable.

2.2.2 Omitting NAs vs Data Imputation

If we can omit the NAs or if it may be necessary to impute the missing data points, depends on the type of missingness. If data is missing completely at random (MCAR), we can omit the NAs, if it is just missing at random (MAR) we would rather impute the data. If the data is missing not at random (MNAR), it would be a quite difficult problem because we cannot easily impute the missing data then. To find out if we can just omit the data, an MCAR test was applied.

We test the target-related variables q25 to q29 for potential pattern(s) in the missing data. This results in a p-value of 0, which means we can say for sure, that the data is not missing completely at random. Just omitting all NAs could be problematic and lead to bias.

```r
result <- mcar_test(RISK[, c("q25", "q26", "q27", "q28", "q29")])
print(result)
```

```
## # A tibble: 1 x 4
##   statistic    df p.value missing.patterns
##       <dbl> <dbl>   <dbl>            <int>
## 1      452.    68       0               29
```

Because of this, we will use a rule base approach to create the target variable and impute the predictive variables afterwards. To ensure a good imputation, we need to impute our NAs before reducing the dataset to 2000 observations. To run the imputation properly we need to factorize our nominal and ordinal variables first.

2.3 Target Variable

As a target variable, we decided to calculate a score from 5 questions that reflect the suicide risk of the person (observation) in question. This score is aggregated with a rule-based approach.

After creating the target variable we need to exclude the variables q25 to q29, which were used for creating it, from our dataset. After originally starting with 5 classes (no risk, low risk, moderate risk, high risk, very high risk) for our target variable we reduced it to 3 classes (no risk, low or moderate risk, high risk).

2.4 Imputation

2.5 Reducing and balancing the dataset to 2000 observations

We need to reduce our data to the maximum allowed size of 2000 observations. To ensure the best possible data quality, we want to ensure that our dataset is balanced. Intuitively, we are considering if it is best to still use as much of the non-imputed data for our smaller dataset as possible, before filling it up with imputed data, since non-imputed data is usually of better quality. On the other hand the data seemingly shows patterns in the missingness so there are reasons to just do a stratified sampling over the imputed data as well. To do a proper statified sampling we need to identify the stratification variables. Therefore we will calculate the correlations with the target variable and see which variables are highly correlated to our target variable. These will then as well as the target variable be used as stratification variables.

2.6 Simple Synopsis of the Dataset

number of observations: 2000 number of variables: 100

datatypes: factor: 96 - nominal variables: 29 - ordinal variables: 67 numeric variables: 4 - discrete variables: 96 (here all factor variables) - continuous variables: 4 (here all numeric variables)

3. Additional Data Preparation

question: would it introduce information leakage to reduce the features before splitting the data?

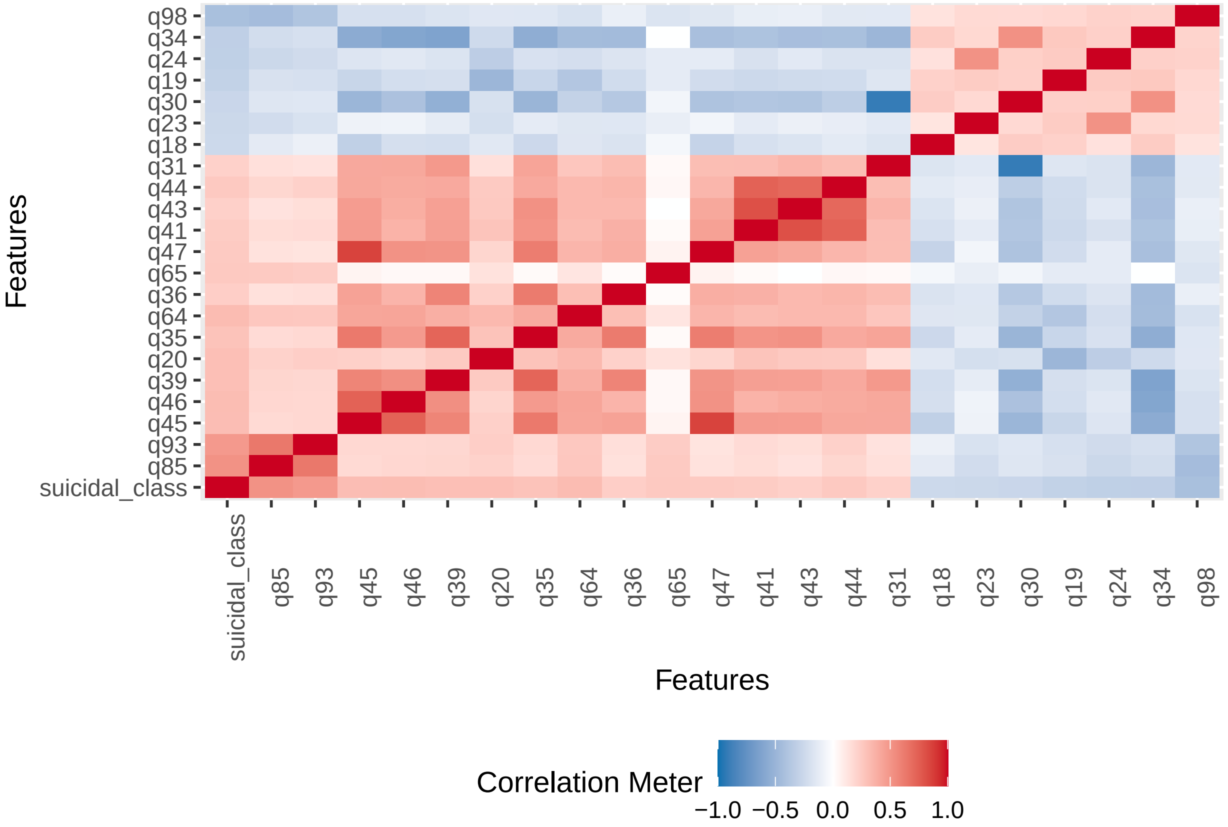
3.1 Feature Reduction

Since our dataset has lots of variables, we decided to start by excluding some variables depending on the estimated feature importance.

3.1.1 Correlations Unfortunately at this point we have to many variables to do a pairs plot or correlation plot with a visually usable outcome. We will therefore perform a correlation analysis only with respect to the target variable and in numeric format instead of any visual plot.

```
## Rows: 1998 Columns: 85
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (2): q6orig, q7orig
## dbl (83): raceeth, q1, q2, q3, q6, q7, q8, q9, q10, q11, q16, q18, q19, q20,...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
## Warning in lapply(RISK_reduced, as.numeric): NAs introduced by coercion
```



```r
print(corr_vars)
```

```
##  [1] "suicidal_class" "Prob"           "q85"            "q93"
##  [5] "q45"            "q46"            "q39"            "q20"
##  [9] "q35"            "q64"            "q36"            "q65"
## [13] "q47"            "q41"            "q43"            "q44"
## [17] "q31"            "q18"            "q23"            "q30"
## [21] "q19"            "q24"            "q34"            "q98"
## [25] "Stratum"
```

The 18 variables with high correlations (>0.25) with our target variable are: q93
q85
q45
q35
q36
q39
q64
q46

q20
q30
q24
q19
q98
q34 q41 q43 q44 q47

3.1.2 Feature Selection Algorithm Since the data still has a lot of variables, we need to use a feature selection technique to reduce the features before using a machine learning method. We chose to use model agnostic methods, because the feature selection should be valid for all methods that are later compared. In an earlier step the variables most correlated with the target variable were already identified. Unfortunately this captures only linear monotonous relationships in the data and does not work well for our nominal categorical features.

[maybe delete chi^2 test + text] We will also use a Chi^2 test between our variables and our target variable to assess their relationship with regards to independence. The variables that are found to have a significant relationship (p >= 0.05 %) with the target are kept.

To capture non-linear relationships as well, information gain between the target and the predictor variables is measured as well. The variables with high information gain with respect to the target variable are kept, because they can contribute more in predicting the target variable.

3.1.3 Feature selection with chi^2

3.1.4 Information gain for feature selection

3.1.5 Domain knowledge for final feature selection

```
## [1] "q7orig" "q40"
```

Let's see what those variables actually stand for. "q7orig" and "q6orig" cannot be found in the data manual and will therefore be discarded. According to the data manual, "PSUs consist of counties, groups of smaller adjacent counties, or sub-areas of very large counties. "PSU" indicates the PSU the school the student attends was assigned to." (p.14). It is possible, that the district/locality of a school plays a role in the risk of suicide. For example for queer students in a very religious place. Q22 is the variable that describes physical dating violence. Therefore q22 is also a valid choice as a predictor variable for our suicidal score target variable. Q40 encodes the range of age when a student first got into contact with drinking alcohol. This might be an indicator for a negligent social surrounding if someone is exposed to an alcoholic drink in an early age and therefore also could be a valid predictor variable in our case.

3.2 Naming the variables and the factor levels

For an easy understanding of their values, the variables and levels are named according to their content. This is an optional step. It can lead to a better readability of our data frame though.

3.3 Splitting the Data

According to the project requirements we split our data in 60% Training, 20% Validation and 20% Testing Data. Since we want to do a cross validation we will split into 80% Training and 20% Testing Data.

4. Machine Learning Models

-> not linearly separable -> chose models that are good in separating non-linear relationships -> chose model preferably from ML2 (at least 1) -> chose a model preferable with results that can be visualized

SVM and Naive Bayes

4.1 Short Mathematical Overview on the used Methods

4.1.1 Naive Bayes Classification

Ideally a classifier is able to detect the class k which maximizes the conditional probability $P(Y = k|X = x_1, ..., x_p)$. A Bayes Classifier would calculate these probabilities for each of the classes exactly, but usually it is only possible to approximate those. The Naive Bayes Classifier is one method of approximation. It approximates by “naively” assuming the conditional independence of predictor variables. This leads to simpler calculations. The joint probability of two events A and B $P(A \cap B) = P(A \mid B) * P(B)$ can be simplified to $P(A \cap B) = P(A) * P(B)$ under the independence assumption, since conditional independence means $P(A \mid B) = P(A)$. The conditional probabilities of a class k can be calculated with the Bayes Theorem. It states that:

$$P(k \mid X) = \frac{P(X \mid k) \cdot P(k)}{P(X)}$$

Since the denominator only uses our predictor variables, we only need to focus on the nominator and find the maximum class for each observation to be classified. We can express this relationship with the proportionality operator:

$$P(k|X) \propto P(X|k) \cdot P(k)$$

At this point, the assumption of independent predictor variables simplifies the calculations if there is more than one predictor variable. Instead of calculating $P(k|x_1, ..., x_p) \propto P(x_1, ..., x_p|k) \cdot P(k)$ where $P(x_1, ..., x_p|k)$ is quite complicated to calculate because of all the possible dependencies among the variables, the independence assumption leads to:

$$P(k|x_1, ..., x_p) \propto P(k) \cdot \prod_{i=1}^{p} P(x_i|k)$$

Often this yields good results even if the quite strong assumption of conditional independence is not met. If the dependencies do not contribute that much to the outcome, the approximation is still quite good.

4.1.2 Support Vector Machines (SVM)

The name Support Vector Machines already describes some elements of this method. A certain number of data points will define the (linear) boundary between two classification regions, these are called the support vectors. The support vectors are the datapoints (observations) that lie closest to our decision boundary. The boundary in two dimensional space is a line, in three dimensions a plane and in more than three dimensional space a hyperplane. For our dataset, we need a multidimensional hyperplane. The number of dimensions depend on the number of our predictor variables. We need to find the hyperplane, that separate our data into the classes of our target variable best. The best hyperplane is the one that maximizes the margin between the support vectors of the different classes. The margin is a strip on each of the boundaries sides. In the case of a hard classifier this strip does not contain any points. But we will have a soft classifier with the cost C as a hyperparameter. This cost C describes a budget that we allow for points within the margin or on the other side of the boundary. Depending on the position of the point, the amount it attributes to the total cost changes. A point on the correct side of the margin will not attribute to the total cost at all. If it is in the margin, but on the correct side of the boundary it will attribute between zero and one, if on the wrong side of the boundary but within the margin it will attribute with one to two and if on the wrong side of the margin it will cost more than two. Mathematically we need to maximize the Margin M with respect to α_0, α_i and ϵ_i in the following objective function:

$$y_i \left(\alpha_0 + \sum_{j=1}^{n} \alpha_i K(\mathbf{x}_i, \mathbf{x}_j) \right) \geq M(1 - \epsilon_i)$$

The following constraints are given: $\sum_{i=1}^{n} \alpha_i^2 = 1$, $\epsilon_i \geq 0$ and $\sum_{i=1}^{n} \epsilon_i < C$ with C ≥ 0 and $i = 1, ..., n$

For our SVM approach, we will try different kernels K as hyperparameters:

A linear Kernel

$$K(u, v) = \langle \mathbf{u}, \mathbf{v} \rangle = \sum_{j=1}^{p} u_j v_j$$

a polynomial Kernel

$$K(u, v) = (c + \langle \mathbf{u}, \mathbf{v} \rangle)^d, \; with \; d > 1$$

and a radial Kernel

$$K(u, v) = \exp\left(-\gamma \sum_{i=1}^{p} (u_i - v_i \right)$$

4.2 Preprocessing

svm in R tutorial + nice visualization: https://www.datacamp.com/tutorial/support-vector-machines-r
We scale the data as part of pre-processing. Scaling transforms the data to have unit variance, further contributing to uniformity across different scales and improving algorithm performance.

4.3 Hyperparameter Optimization

Hyperparameter Optimization (HPO) enables the testing of various hyperparameter combinations to identify the optimal settings that maximize our target evaluation metric, namely the model's accuracy.
The grid search method allows for the exhaustive pairing of each hyperparameter with every other, albeit at a significant computational cost. This approach, however, offers the advantage of explicitly specifying the values for testing.
Additional to scaling we also use centering as a part of preprocessing. Centering the data ensures that each feature has a mean of zero. This is particularly useful when features are on different scales and can improve the performance by removing bias due to the scale of the features.

4.3.1 HPO of Naive Bayes

4.3.2 HPO of SVM

For the SVM we examine three distinct kernel types - linear, radial, and polynomial - each characterized by unique parameters, in addition to the common cost parameter.
Kernel and the cost parameter C have a significantly impact on the model's performance and need to be tuned carefully.
Additionally to scaling we're also centering the data, which involves subtracting the mean from each feature, ensuring that each feature has a mean of zero. This is particularly useful when features are on different scales and can improve the performance of support vector machines by removing bias due to the scale of the features. For programming purposes we use the build-in preProcess parameter of the train function where the training data is automatically scaled and centered.
?A large value of C leads to … being heavily penalised so there are few points in the margin/missclassified (Lecture 7, p.5)

Given the list of all resulting models, coming from the HPO of the SVM, we extract the best model for further inspection and results using the Accuracy as our focused metric.

5. Comparison of the Models / Model's Performance on Test Data

if one model performs better, is this improvement significant to a usual significance level?

5.1 Quantitative

5.1.1 Confusion Matrix

###### 5.1.1.2 Naive Bayes
5.1.1.2 SVM

```
##   [1] 1 2 3 3 2 2 3 3 3 2 2 2 3 2 3 1 1 3 2 2 2 3 2 2 2 3 2 3 1 1 3 3 3 3 3 3 3
##  [38] 3 3 3 2 1 3 3 2 1 2 2 2 3 3 2 2 3 3 2 3 3 2 2 3 2 2 2 2 1 3 3 3 3 3 2 3 2
##  [75] 2 2 2 3 2 3 2 3 1 3 1 2 3 2 2 2 3 2 1 3 1 3 3 2 2 2 3 2 1 2 3 3 2 2 2 3 2
## [112] 2 3 3 2 3 3 3 3 2 3 3 3 1 3 2 2 3 3 3 3 1 2 2 1 2 1 1 3 2 1 2 2 2 3 2 2 2
## [149] 2 3 3 2 1 1 3 1 2 2 2 1 2 1 2 3 3 3 2 3 1 1 2 2 3 2 1 1 2 2 1 1 2 2 2 3 2
## [186] 2 2 3 2 1 2 2 1 2 2 2 2 2 3 2 2 1 3 2 2 2 1 2 3 2 2 2 3 3 3 1 2 3 1 1 2 1
## [223] 2 2 3 2 1 3 2 2 2 3 3 2 2 1 2 2 1 1 2 3 1 3 2 1 2 2 2 2 1 3 3 3 2 3 2 2 1
## [260] 1 3 3 2 3 2 3 2 2 1 1 1 1 3 1 1 1 1 1 2 2 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1
## [297] 1 1 1 1 1 1 2 1 1 1 1 1 1 2 2 1 2 3 1 2 2 1 3 1 1 3 2 1 1 1 1 1 2 1 1 1 1
## [334] 1 2 1 1 3 2 1 1 1 1 1 1 1 1 1 1 1 1 2 1 2 1 2 1 1 1 2 1 2 1 1 1 1 1 1 1 2
## [371] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 2 1 1 1 2 1 2 1
## Levels: 1 2 3
```

```
##           Actual
## Prediction   1   2   3
##          1 106  32  15
##          2  23  69  56
##          3   5  33  63
```

```
## Confusion Matrix and Statistics
##
##                     svm_actual
## svm_best_model_pred   1   2   3
##                   1 106  32  15
##                   2  23  69  56
##                   3   5  33  63
##
## Overall Statistics
##
##                Accuracy : 0.592
##                  95% CI : (0.5422, 0.6405)
##     No Information Rate : 0.3333
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3881
##
##  Mcnemar's Test P-Value : 0.006084
##
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3
## Sensitivity            0.7910   0.5149   0.4701
## Specificity            0.8246   0.7052   0.8582
## Pos Pred Value         0.6928   0.4662   0.6238
## Neg Pred Value         0.8876   0.7441   0.7641
## Prevalence             0.3333   0.3333   0.3333
## Detection Rate         0.2637   0.1716   0.1567
## Detection Prevalence   0.3806   0.3682   0.2512
## Balanced Accuracy      0.8078   0.6101   0.6642
```

Given the Confusion Matrix we can see a Accuracy of 1 meaning the predictions are 100% correct. In this context, a p-value of less than 2.2e−16 for the hypothesis that “Accuracy is greater than No Information Rate” suggests that there is extremely strong statistical evidence that the accuracy of the model is better than what would be achieved by always predicting the most frequent class. This implies that the model has predictive power beyond mere chance and is effectively learning from the features in the dataset.

5.1.2 Precision, Recall and F1-Score

###### 5.1.2.1 Naive Bayes
###### 5.1.2.2 SVM

```
## Recall:   1.24626865671642
## Precison: 1.40522875816994
## F1:       1
```

5.2 Qualitative

—> Ari: maybe delete this section?

5.2.2 SVM without scaling

5.3 Overfitting

Overfitting is a frequent problem in machine learning. It happens when a model learns the training data too well, including all its quirks and noise. As a result, its ability to generalize weakens. When the model is tested with new data, its performance often drops significantly. This is because it’s overly tuned to the training data and doesn’t adapt well to new, unseen data.

5.3.1 Naive Bayes

5.3.2 SVM

By employing Cross-Validation (CV) and utilizing a range of performance metrics such as recall, precision, and F1-score, we try to avoid overfitting in our model. Additionally, our experiments with different kernels reveal that the linear kernel maintains robust performance, even when compared to the more complex radial and polynomial kernels. Prior to modeling, we also took the precaution of reducing the number of features, which further diminishes the risk of overfitting. Moreover, setting the cost parameter C to a moderate level – in our case, 1, which is the default value – lessens the likelihood of overfitting. Considering these factors collectively, we are confident that our model is unlikely to overfit on our training data.

6. Visual Representation

7. Final Discussion

8. References

[1] https://www.cdc.gov/healthyyouth/data/yrbs/overview.htm

[2] https:
//www.cdc.gov/healthyyouth/data/yrbs/pdf/2021/2021_YRBS_Data_Users_Guide_508.pdf