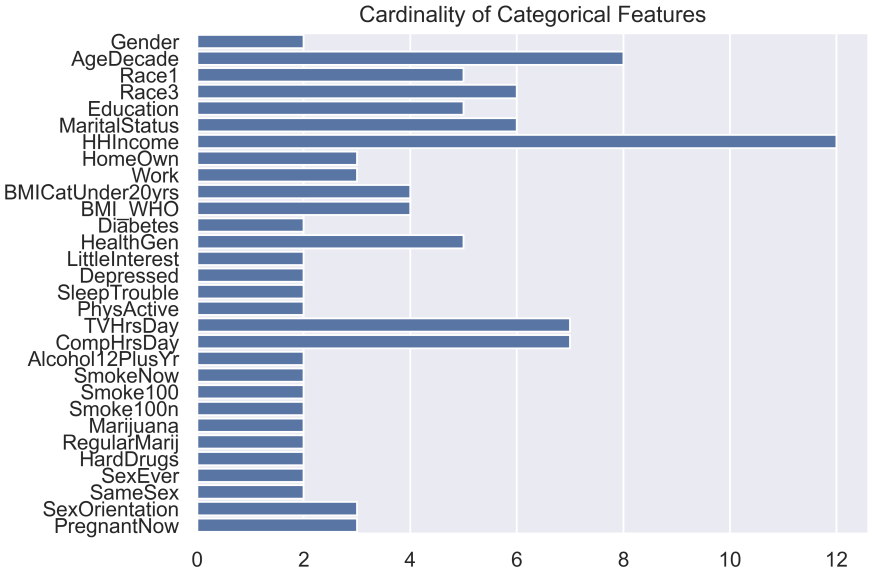
Machine Learning: Project 1

Johannes Misensky, Dejan Prvulovic & David Deket

2023-12-29

Exploratory Data Analysis

Our first step was to explore the data, e.g. shape of the data, column types, missing values, etc. The dataset contains 8000 rows and 71 columns. 41 columns are numeric, 30 are categorical. Most categorical columns are binary, but some have more than two categories. The highest cardinality is 12, which means that one-hot encoding is feasable and won't lead to too many new columns.



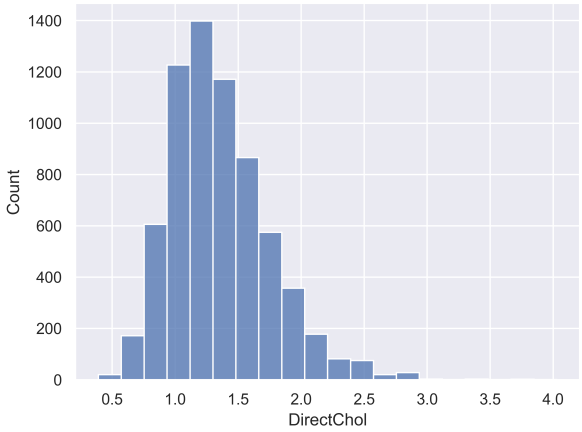
Next, we have a look at the missing values. 26 columns have at least 4000 missing values, which is 50% of the data. Usually, we would impute missing values with the median or mean, however with so many missing values, this would lead to a lot of bias. Instead, we drop all columns with more than 1500 missing values. This leaves us with 28 columns. Afterwards, we remove all rows with missing values in the target variable.

This is the final dataset:

N	Column	Non-Null	Dtype	N	Column	Non-Null	Dtype
0	NR	6782	int64	14	Pulse	6414	float64
1	Gender	6782	object	15	BPSysAve	6404	float64
2	Age	6782	int64	16	BPDiaAve	6404	float64
3	AgeDecade	6540	object	17	BPSys1	6154	float64
4	Race1	6782	object	18	BPDia1	6154	float64
5	HHIncome	6252	object	19	BPSys2	6262	float64
6	HHIncomeMid	6252	float64	20	BPDia2	6262	float64
7	Poverty	6310	float64	21	BPSys3	6268	float64
8	HomeRooms	6733	float64	22	BPDia3	6268	float64
9	HomeOwn	6735	object	23	DirectChol	6782	float64
10	Weight	6731	float64	24	UrineVol1	6729	float64
11	Height	6737	float64	25	UrineFlow1	6305	float64
12	BMI	6727	float64	26	Diabetes	6782	object
13	BMI_WHO	6705	object	27	PhysActive	6284	object

Regression

The target variable `DirectChol` for the regression has the following distribution:



Our pre-processing pipeline for the regression task consists of the following steps:

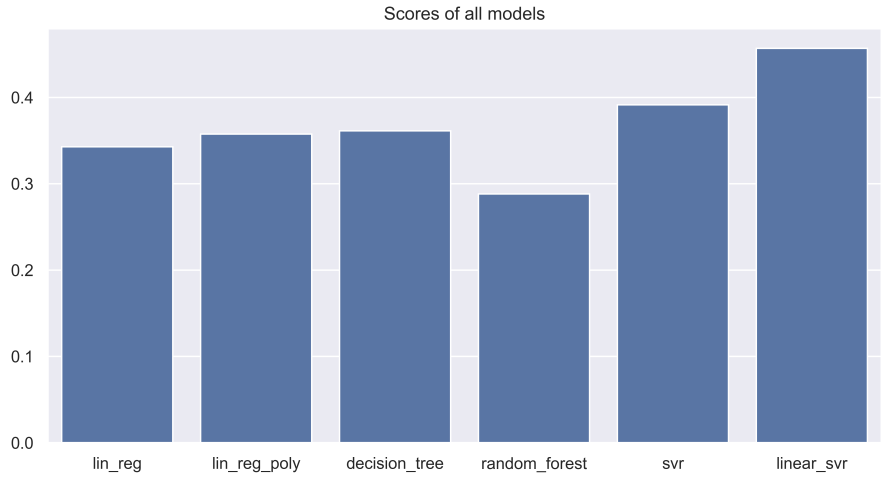
1. Drop columns with more than 1500 missing values
2. For numerical columns, impute missing values with the median and use standard scaling

3. For categorical columns, use one-hot encoding with `handle_unknown='ignore'` to avoid errors when encountering unknown categories in the data

We use the following models for the regression task:

- Linear Regression (with and without polynomial features)
- Decision Trees
- Random Forests
- Support Vector Regressions (with and without kernel trick)

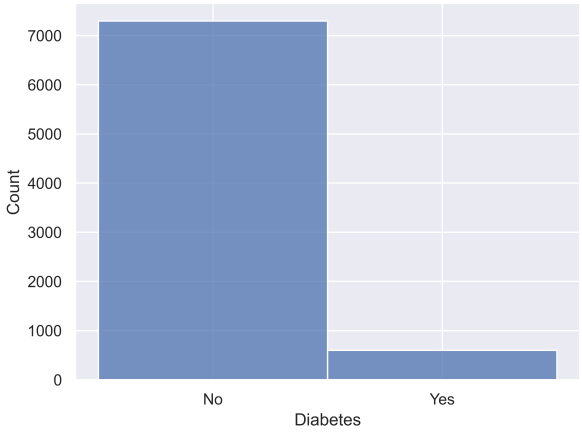
Models that have adjustable hyperparameters are trained using 5-fold cross-validation and scored on root mean squared error (RMSE).



The best model is the Random Forest with an RMSE of about 0.30. Interestingly, the linear regression models perform second best, indicating that the relationship between the features and the target variable is approximately linear and that data is more important than model choice. The worst model is the linear support vector regression, though the SVR with the kernel trick still performs worse than the linear regression models.

Classification

The target variable `Diabetes` for the classification has the following distribution:



Clearly, a class imbalance making the classification task more difficult. Therefore, we upsample the minority class by randomly duplicating rows until both classes have the same number of rows. There are more advanced techniques to deal with class imbalance (e.g. SMOTE), but we don't use them here in order to save another dependency.

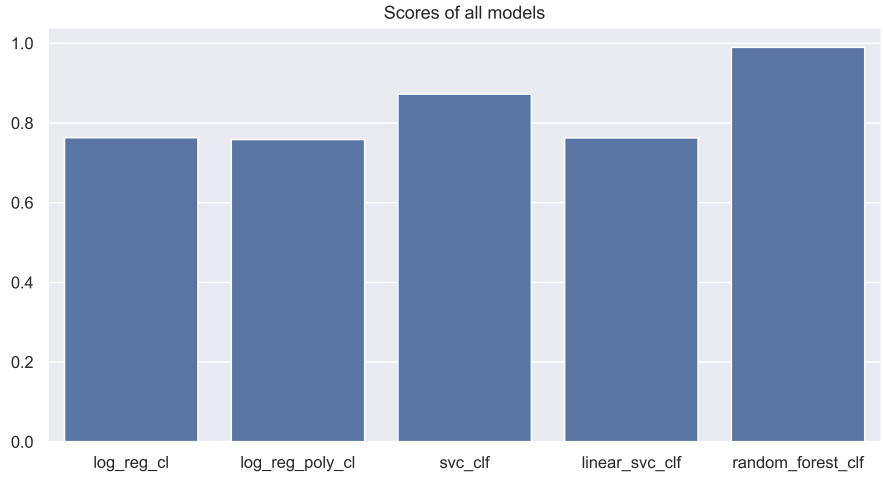
Our pre-processing pipeline for the classification task consists of the following steps (slightly different from the regression task to try out different techniques):

1. Drop columns with more than 1500 missing values
2. For numerical columns, impute missing values with the median, apply min-max scaling and PCA
3. For categorical columns, use one-hot encoding with `handle_unknown='infrequent_if_exist'` and set `sparse_output=True` to avoid memory errors

We use the following models for the classification task:

- Logistic Regression (with and without polynomial features)
- Support Vector Classifiers (with and without kernel trick)
- Random Forest

Models that have adjustable hyperparameters are trained using 5-fold cross-validation and scored on the weighted F1 score.

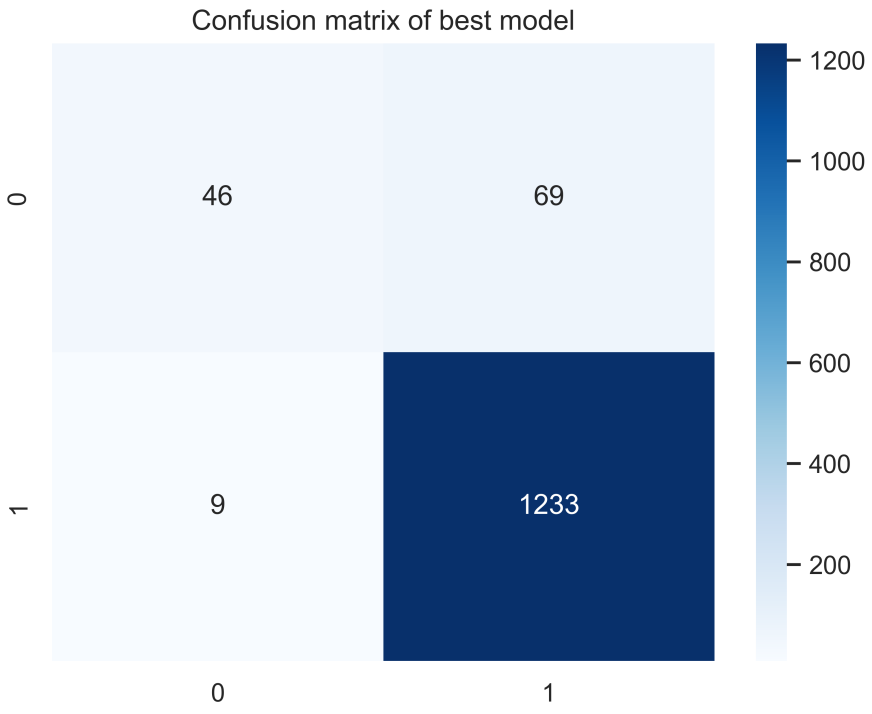


The best model is again the Random Forest with an F1 score of close to 1. This time, the second best model is the support vector classifier, whereas the linear support vector classifier performs worst (kernel trick definitely helps). The logistic regressions with regularization perform similar to the SVCs, but the polynomial features don't help.

Lastly, we have a look at the confusion matrix of the Random Forest on the test set:

Pred	precision	recall	f1-score	total
Diabetes	0.84	0.40	0.54	115
No Diabetes	0.95	0.99	0.97	1242
accuracy			0.94	1357
macro avg	0.89	0.70	0.76	1357
weighted avg	0.94	0.94	0.93	1357

No diabetes is predicted correctly most of the time, which is easy to achieve given the class imbalance. Diabetes is predicted correctly 84% of the time (46 out of 46+9=54 true diabetes cases), however the model suffers from a low recall of 40%. This means we get many false positives (46 cases are truely diabetes from the predicted 46+69=115 cases), which probably is not ideal for a medical diagnosis algorithm.


Confusion matrix of best model
46
69
9
1233
0
1
0
1
1200
1000
800
600
400
200