

Machine Learning - Bericht zum Abschlussprojekt

Fallstudie: Diabetes, Bearbeitungszeitraum 28.04. - 02.05.2025

1. Teilnehmer und Arbeitsschwerpunkte

Name	Bereiche
Emilia Annacker	NaiveBayes
Isabel Barth	Support Vector Machine
Marc Engelmann	Random Forest
Christopher Puschies	Logistische Regression

2. Aufgabenstellung

a) Zielsetzung der Studie

Im Auftrag des Gesundheitsamts analysieren wir Gesundheits- und Verhaltensdaten, um die Entstehung von Diabetes besser zu verstehen und Risikofaktoren frühzeitig zu identifizieren. Ziel der Analyse ist es, ein datenbasiertes Fundament für ein Präventions- und Frühinterventionsprogramm zu schaffen, das Patienten mit manifestem Diabetes oder einem hohen Diabetesrisiko zuverlässig erkennt.

Durch gezielte Früherkennung sollen frühzeitig Behandlungspfade eingeleitet, die Gesundheit der Bevölkerung nachhaltig verbessert und Folgekosten durch diabetische Komplikationen im Gesundheitssystem deutlich reduziert werden.

Beschränkungen: Die Studie findet unter starkem Zeit- und Budgetdruck statt. Erste Ergebnisse werden innerhalb von 3 Tagen erwartet. Monetäres Zusatzbudget ist nicht vorgesehen.

b) Anforderungen und zentrales Gütekriterium

Bei der Modellierung legen wir besonderes Augenmerk auf eine hohe Sensitivität, um möglichst alle Risikopatienten zu erfassen und gefährliche Fehldiagnosen (False Negatives) zu vermeiden. Eine ausgewogene Präzision bleibt weiterhin wichtig, tritt jedoch hinter der Maximierung der Sensitivität

zurück, da die gesundheitlichen und wirtschaftlichen Folgen von übersehenen Risikopatienten die Kosten von falsch-positiven Befunden deutlich übersteigen.

Da die Modellergebnisse in der ärztlichen Praxis nachvollziehbar und kommunizierbar sein müssen, ist die Interpretierbarkeit ein zentrales Kriterium bei der Auswahl der Algorithmen.

Modelle mit hoher Erklärbarkeit (wie z. B. logistische Regression, Entscheidungsbäume, Bayesschätzer) werden bevorzugt, sofern sie eine ausreichend hohe Vorhersagegenauigkeit liefern. Erst wenn diese Algorithmen keine verlässlichen Vorhersagen erreichen, wird diese Anforderung zu Gunsten der Vorhersagequalität vernachlässigt.

Demonstrierend wird Recall als zentrales Gütekriterium mit einem Mindestmaß von 0.9 festgelegt.

3. Beschreibung der Daten


a) Bemerkungen zur Datenquelle

Als Datenquelle wurde der *kaggle* Datensatz: "diabetes health indicators dataset" verwendet:

(<https://www.kaggle.com/datasets/alexteboul/diabetes-health-indicators-dataset/data>)

Die Datenbeschaffung wurde mit Hilfe des oben stehenden Links durchgeführt und war mit wenigen Schwierigkeiten verbunden. Einzig vor der konkreten Auswahl des Datensatzes war eine kurze Diskussion notwendig. Von den vorliegenden Datensätzen...

 diabetes_012_health_indicators_BRFSS2015.csv

 diabetes_binary_5050split_health_indicators_BRFSS2015.csv

 diabetes_binary_health_indicators_BRFSS2015.csv

...wurde der zweite Datensatz

"diabetes_binary_5050split_health_indicators_BRFSS2015.csv" verwendet.

Dieser Datensatz eignet sich aufgrund seiner ausbalancierten Daten gut, um zügig wichtige Merkmale zu identifizieren und Modelle zu erstellen.

Der Hauptnachteil ist, dass dies nicht die Realität widerspiegelt - in Wirklichkeit gibt es mehr Nicht-Diabetiker als Diabetiker - und dass somit das hiermit trainierte Modell bei der Anwendung in der Praxis schlechtere Werte ergeben wird als ein auf einem nicht-balanciertem Datensatz trainiertes Modell.

b) Meinungsbildung auf Basis von Expertenwissen (Internet-Recherche)

Internet-Recherche zur Frage: Was ist relevant für Diabetes?

Es wurde eine Recherche zu den möglichen Risikofaktoren von Diabetes durchgeführt. Folgende Quelle wurden zu Rate gezogen: ChatGPT, sowie:

https://www.niddk.nih.gov/health-information/diabetes/overview/risk-factors-type-2-diabetes?utm_source=chatgpt.com

Die folgenden Faktoren gehören laut medizinischer Forschung zu den Hauptursachen für Diabetes:

- Übergewicht/Adipositas (extrem hoch),
- körperliche Inaktivität (sehr hoch),
- ungesunde Ernährung (hoch) - insbesondere (zu) viel Zucker
- genetische Prädisposition (hoch),
- Alter (ab 45 Jahren, hoch),
- Bluthochdruck (mittelhoch) und
- Dyslipidämie (schlechte Blutfettwerte, mittelhoch)."

Insofern in Anbetracht der verfügbaren Daten möglich, sollen die oben genannten Erkenntnisse in der Datenauswahl und -Analyse berücksichtigt werden. Zusätzlich werden Variablen in die Analyse aufgenommen, die aufgrund einer deskriptiven Analyse eine hohe Relevanz für die Vorhersage von Diabetes wahrscheinlich ist. Auf Basis unserer Recherche werden die folgenden Hypothesen formuliert:

H1:

Patienten mit höherem BMI haben ein erhöhtes Risiko, an Diabetes zu erkranken.

→ Motivation: Übergewicht (insbesondere viszerale Adipositas) ist der stärkste Einzelfaktor laut Forschung.

H2:

Patienten mit Bluthochdruck und/oder Dyslipidämie weisen eine höhere Diabeteswahrscheinlichkeit auf als Patienten ohne diese Vorerkrankungen.

→ Motivation: Beide Faktoren sind Teil des „metabolischen Syndroms“, das stark mit Diabetesentwicklung verbunden ist.

H3:

Mit zunehmendem Alter steigt das Risiko einer Diabeteserkrankung signifikant an.

→ Motivation: Insulinsensitivität sinkt mit dem Alter, auch unabhängig von Gewicht.

H4:

Körperliche Inaktivität ist ein signifikanter Prädiktor für das Vorliegen oder die Entwicklung von Diabetes.

→ Motivation: Bewegung ist einer der stärksten Schutzfaktoren gegen Insulinresistenz.

H5:

Ein Machine-Learning-Modell, das auf den wichtigsten Prädiktoren sowie zusätzlich durch eine deskriptive Analyse identifizierten relevanten Merkmalen trainiert wird, erreicht bei der Vorhersage von Diabetesfällen eine Recall-Rate von mindestens 90 %

Eine weitere intensive Auseinandersetzung mit den Daten, der inhaltlichen Relevanz und der Codierung wäre notwendig und empfehlenswert, wird aber vom Projekt-Team an dieser Stelle aus Zeitgründen nicht weiter vorgenommen.

c) Operationalisierung und Inhalte der Spalteninformation

Die Merkmale des Datensatzes basieren auf Fragen aus einer jährlichen Umfrage des Behavioral Risk Factor Surveillance System (BRFSS) des Centers for Disease Control and Prevention, einer Regierungsorganisation in den USA. Ursprünglich besteht dieser Fragebogen auf wesentlich mehr Fragen als im Datensatz widergespiegelt und ist nicht auf Diabetes-Risikoverhalten beschränkt (siehe:

<https://www.cdc.gov/brfss/questionnaires/pdf-ques/2015-brfss-questionnaire-12-29-14.pdf>).

In der Umfrage wurden viele Antworten differenzierter abgefragt, beispielsweise bei Cholesterin-Tests, Gesundheitsstatus oder Alter. Im Kaggle-Datensatz wurden diese Antworten jedoch meist binär zusammengefasst (z.B. Cholesterin-Test: Ja/Nein) oder in Kategorien eingeteilt (z.B. Altersgruppen statt exaktem Alter).

Spaltenname	Typ und Anzahl	Beschreibung
Diabetes_binary	Kategorisch (2 Werte)	Gibt an, ob die Person an Diabetes leidet (1 = Ja, 0 = Nein)
HighBP	Kategorisch (2 Werte)	Vorgeschichte von Bluthochdruck (1 = Ja, 0 = Nein)
HighChol	Kategorisch (2 Werte)	Vorgeschichte von hohem Cholesterinspiegel (1 = Ja, 0 = Nein)
CholCheck	Kategorisch (2 Werte)	Cholesterinkontrolle in den letzten 5 Jahren (1 = Ja, 0 = Nein)
BMI	Numerisch (viele Werte)	Body-Mass-Index (Gewicht/Größe²)
Smoker	Kategorisch (2 Werte)	Hat mindestens 100 Zigaretten im Leben geraucht (1 = Ja, 0 = Nein)
Stroke	Kategorisch (2 Werte)	Vorgeschichte eines Schlaganfalls (1 = Ja, 0 = Nein)
HeartDiseaseorAttack	Kategorisch (2 Werte)	Vorgeschichte von Herzkrankheiten oder Herzinfarkt (1 = Ja, 0 = Nein)
PhysActivity	Kategorisch (2 Werte)	Körperliche Aktivität in den letzten 30 Tagen (außer beruflich) (1 = Ja, 0 = Nein)
Fruits	Kategorisch (2 Werte)	Täglicher Verzehr von Obst (1 = Ja, 0 = Nein)
Veggies	Kategorisch (2 Werte)	Täglicher Verzehr von Gemüse (1 = Ja, 0 = Nein)
HvyAlcoholConsump	Kategorisch (2 Werte)	Starkes Trinken (Männer >14, Frauen >7 Getränke pro Woche) (1 = Ja, 0 = Nein)
AnyHealthcare	Kategorisch (2 Werte)	Verfügt über eine Krankenversicherung oder ähnliche Absicherung (1 = Ja, 0 = Nein)
NoDocbcCost	Kategorisch (2 Werte)	Arztbesuch aus Kostengründen in den letzten 12 Monaten vermieden (1 = Ja, 0 = Nein)
GenHlth	Kategorisch (5 Werte)	Allgemeiner Gesundheitszustand (1 = Ausgezeichnet, ..., 5 = Schlecht)
MentHlth	Numerisch (31 Werte: 0-30)	Anzahl der Tage mit schlechter psychischer Gesundheit in den letzten 30 Tagen
PhysHlth	Numerisch (31 Werte: 0-30)	Anzahl der Tage mit schlechter körperlicher Gesundheit in den letzten 30 Tagen
DiffWalk	Kategorisch (2 Werte)	Schwierigkeiten beim Gehen oder Treppensteigen (1 = Ja, 0 = Nein)
Sex	Kategorisch (2 Werte)	Biologisches Geschlecht (1 = Männlich, 0 = Weiblich)
Age	Kategorisch (13 Werte)	Alterskategorie (1 = 18-24 Jahre, ..., 13 = 80+ Jahre)
Education	Kategorisch (6 Werte)	Bildungsstand (1 = Keine Schule, 2 = Bis zur 8. Klasse, 3 = Einige Jahre High School, kein Abschluss, 4 = High School Abschluss, 5 = Einige Jahre College oder technischer Abschluss, 6 = College-Abschluss)
Income	Kategorisch (8 Werte)	Einkommensklasse (1 = < \$10.000, ..., 8 = > \$75.000)

Abbildung: Übersicht über die Operationalisierung der Variablen

Abschließende Bemerkung zur Operationalisierung der Variablen:

Das Projektteam hätte sich gewünscht, dass mehr Ursprungsinformation bzw. Varianz erhalten geblieben wäre. Zum Beispiel wäre es wünschenswert, wenn "High BP" (hoher Blutdruck) nicht in ja/ nein codiert wäre. Besser wären die tatsächlichen Blutdruck-Werte (auf einer metrischen Skala) gewesen, um so bessere Vorhersagen möglich zu machen. Die Einteilung in Kategorien führt häufig zu künstlichen Grenzen, die einer eigentlich kontinuierlichen Struktur nicht gerecht werden. Dies kann dazu führen, dass Übergänge zwischen den Kategorien als größere Sprünge erscheinen, während gleich große Unterschiede innerhalb einer Kategorie unberücksichtigt bleiben.

d) Untersuchung des Datensatzes

1. Der Datensatz besteht aus 70692 Einträgen und 22 Spalten (6,3 MB)
2. Die Zielspalte (y) ist in Spalte1 und wird in einen separaten Datensatz überführt. (Diabetes Yes/ No)
3. Die Ausprägung Diabetes ja/nein ist mit 35.346 exakt gleichverteilt. Dies entspricht sehr wahrscheinlich nicht der Verteilung von Diabetes in der Grundgesamtheit der Bevölkerung. Diese Auffälligkeit wird aber vorerst nicht weiter berücksichtigt, da dies ein Merkmal des ausgewählten Datensatzes ist:
"diabetis_binary_5050split_health_indicators_BRFSS2015.csv"
4. Der Datensatz enthält 21 beschreibende Features (X).
5. Datenformat aller Einträge ist float. Es gibt allerdings eine Vielzahl von "0 und 1" codierten Variablen (Bernoulli-verteilt)
6. Mit der `pandas.info` Methode wurde auf **fehlende Werte** untersucht. Es gibt keine fehlenden Werte, somit kann auf den Einsatz von Imputern an dieser Stelle verzichtet werden.
7. Die Daten wurden ferner mittels `pandas.unique()` auf **versteckte fehlende Werte** untersucht (wie etwa Zahlen wie '0' oder '-1', die für einen fehlenden Wert stehen). Auch hier konnten keine Auffälligkeiten festgestellt werden. -> **Somit wird kein Imputer verwendet.**
8. Des Weiteren wurden die Daten mit der `pandas.describe()` Methode untersucht. Hierbei kam es zu folgenden Beobachtungen:

- Es liegen 3 (quasi-)metrische Variablen vor: BMI, Mental Health, Physical Health
- Es liegen 4 ordinal skalierte Variablen vor: General Health, Age, Education, Income
 - Es wurde diskutiert, ob diese im weiteren Verlauf in binäre Daten umcodiert werden müssen. Entscheidung: Zunächst werden die Variablen nicht umcodiert. Es bleibt dem Anwender der jeweiligen Algorithmen überlassen, ob eine binäre Umcodierung jeweils notwendig ist.
 - Das Projekt-Team hat sich auch gegen eine One Hot Codierung entschieden, da alle Variablen interpretierbare und sinnvolle Abstände gemäß ihrer Ordnung aufweisen (Bsp. Alter 1="18-24"; 2="25-30" usw.). Für einzelne Algorithmen kann hiervon abgewichen werden, um Nachteile der Kategorisierung zu kompensieren.
- Alle anderen Daten sind kategoriale Daten die mit 0 und 1 kodiert sind.

9. Diskussion des BMI: Dieser zeigt einen max. Wert von 98. Dieser erschien sehr hoch. Eine visuelle Darstellung der BMIs größer 40 zeigt, dass 5000 Datensätze einen extrem hohen BMI aufweisen:

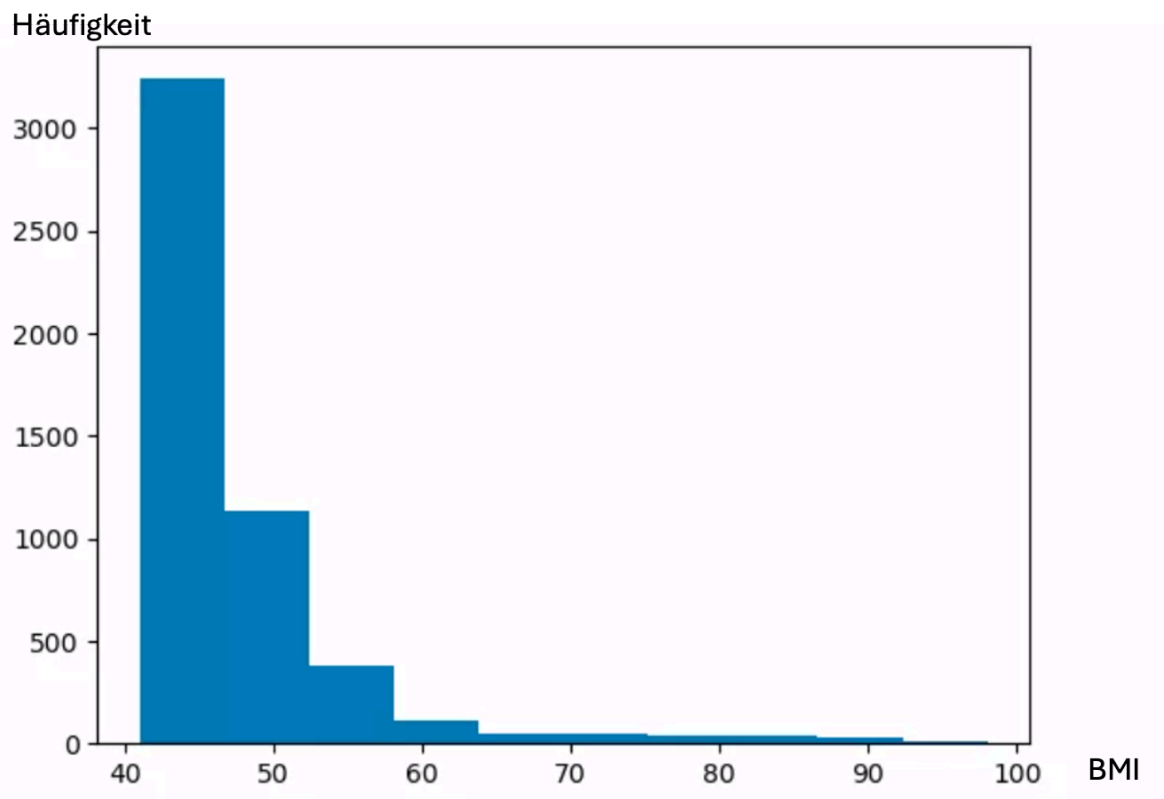


Abbildung: Häufigkeiten von "BMI" über 40

Eine weitere Recherche ergab: "Ein Überleben mit einem BMI von über 40 ist möglich, aber mit gravierenden gesundheitlichen Risiken verbunden. Die medizinische Literatur berichtet von Einzelfällen mit noch höheren BMI-Werten (teilweise über 70), aber diese Menschen

sind schwer krank und ihre Lebenserwartung ist drastisch verkürzt. Ein "maximaler" BMI, mit dem ein Mensch noch leben kann, liegt also irgendwo zwischen 70 und 100, wobei dies absolute Ausnahmen sind und meist nur mit intensiver medizinischer Versorgung möglich ist."

Von daher lässt sich nicht generell ausschließen, dass Datensätze mit diesen "extrem hohen" Werten vorliegen und die Daten werden weiterhin verwendet.

10. Mit der Methode `pandas.duplicated` wurde untersucht, ob Duplikate vorhanden sind. Die Methode identifizierte 1.635 (2,3%) vermeintliche Duplikate.

Leider war keine Spalte mit einem Unique Identifier vorhanden. Aufgrund der Kodierung (in überwiegend bool'sche und ordinale Daten) geht das Projekt-Team davon aus, dass es sich hier um natürliche, weil zufällige, Überschneidungen handelt. Dennoch wurden die doppelten Datensätze mit der drop-Methode entfernt, um zu vermeiden, dass die Voraussagekraft der Testmenge beeinträchtigt wird. Im weiteren wird der reduzierte Datensatz mit 69.057 Einträgen genutzt.

e) Auswahl der Variablen und Untersuchung der Korrelationen

1) Korrelationen zwischen einzelnen Spalten und der Zielspalte

Um neben den durch die Forschungsliteratur als wichtig identifizierten Merkmalen weitere relevante Merkmale zu erkennen, ist es sinnvoll, die Korrelation der einzelnen Spalten bzw. Merkmale mit dem Zielwert zu analysieren (Zielspalte: *Diabetes_binary*).

Eine zunächst erstellte Pearson-Korrelationsmatrix wurde für alle Spalten des Datensatzes berechnet. Dieses Verfahren ist jedoch nur für die wenigen numerischen bzw. metrischen Variablen des Datensatzes sinnvoll, da Pearson-Korrelationen auf kontinuierlichen Werten basieren.

Anmerkung: Interessanterweise zeigten sich tendenziell die wissenschaftlich relevantesten Spalten als jene mit den stärksten Korrelationen, jedoch könnte dies ein Zufallsergebnis sein, da methodisch bedingt andere Merkmale nicht berücksichtigt wurden.

Für die zahlreichen kategorialen und ordinalen Spalten ist eine Pearson- oder Spearman-Korrelationsanalyse aufgrund der Datenstruktur und der fehlenden numerischen Basis weniger aussagekräftig. Hierfür existieren alternative Verfahren wie:

- Chi-Quadrat-Test: Geeignet für rein kategoriale Merkmale.
- Cramer's V: Nützlich, um die Stärke der Assoziation zwischen binären oder nominalen Variablen zu messen.

Diese Verfahren wären besonders relevant für binäre Variablen wie *HighBP* oder *Smoker*. Im Falle ordinaler Variablen wie *GenHlth* sollten die Daten möglicherweise vorab aufbereitet werden, um fehlende Werte zu behandeln oder Kategorien sinnvoll zu ordnen.

Aufgrund von Zeit- und Ressourcengründen haben wir beschlossen, die oben genannten Methoden nicht anzuwenden und stattdessen die Beziehungen der Merkmale zur Zielspalte visuell durch Balkendiagramme zu bewerten. Diese Herangehensweise ermöglicht eine schnelle und intuitive Einschätzung der potenziellen Relevanz einzelner Variablen.

Die folgende Tabellen geben einen Überblick, welche der Merkmale weiterverwendet werden samt Begründung.

Tab. 3.e.1: Merkmale zur Weiterverwendung

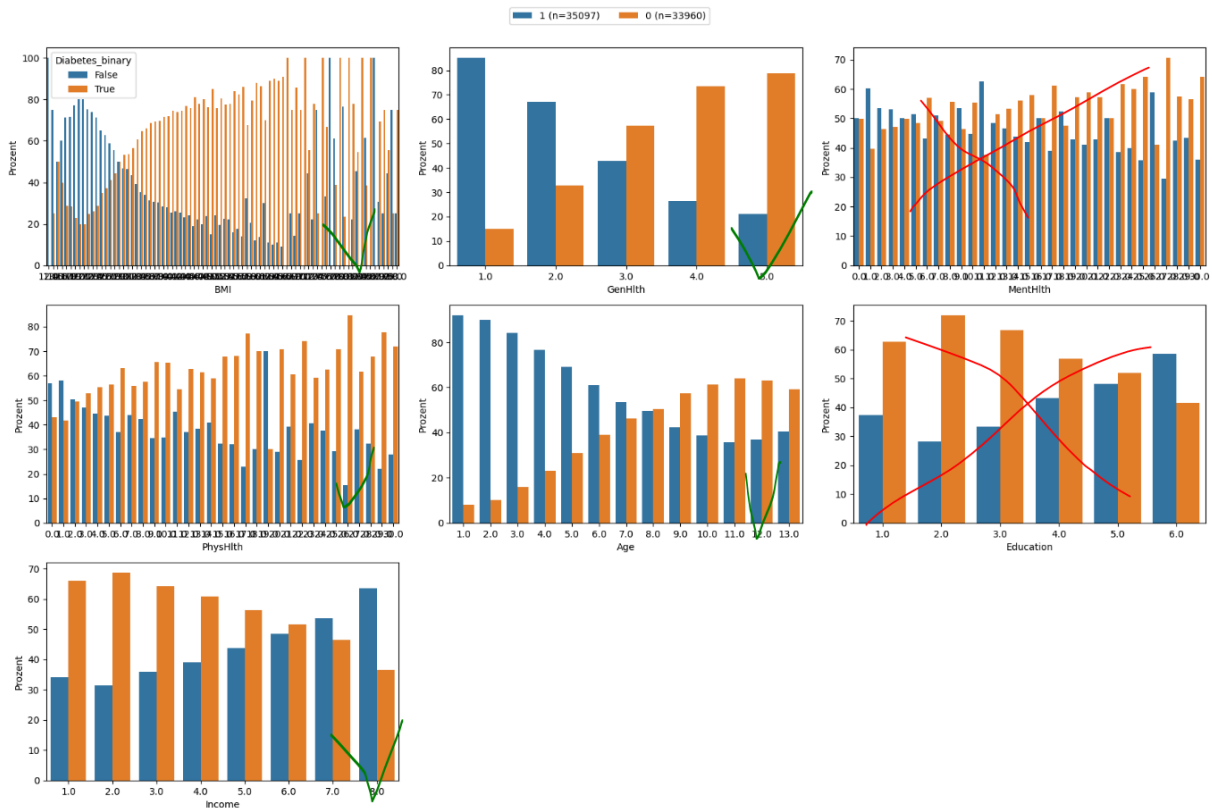
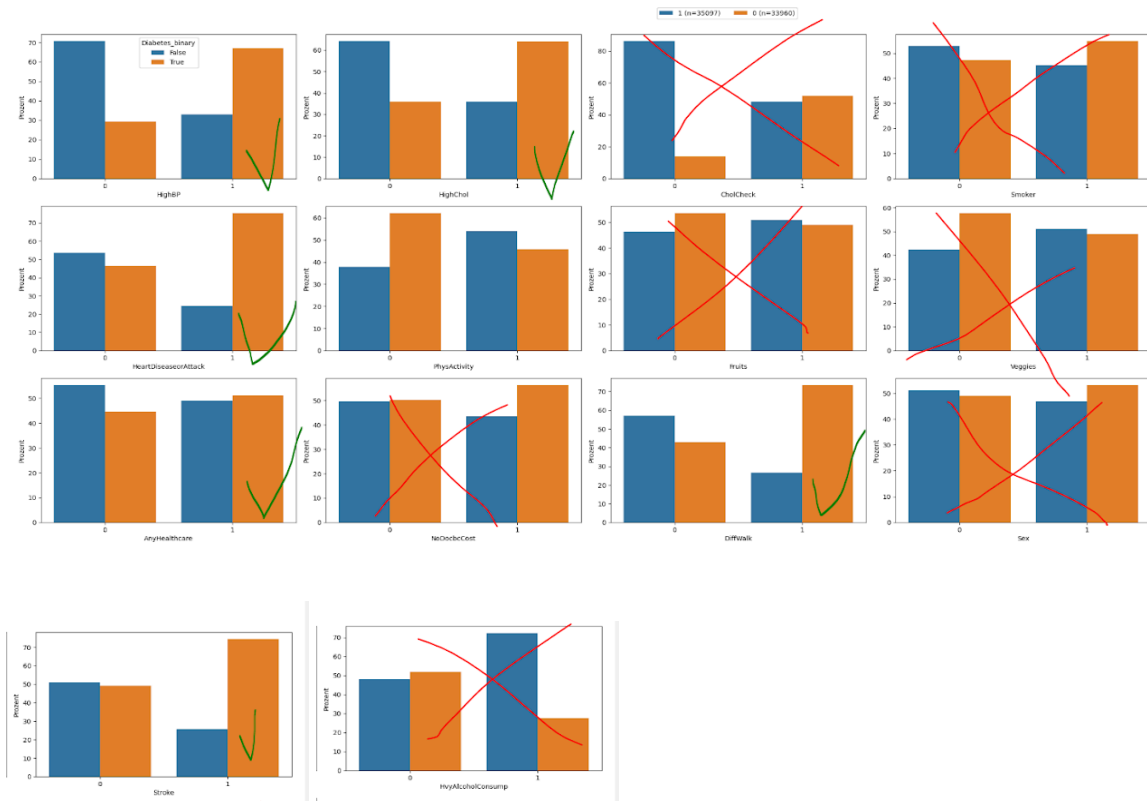
Merkmal	Begründung
BMI	<ul style="list-style-type: none"> + Wissenschaftl. begründete Relevanz (= "sehr hoch"). + Balkendiagramm zeigt starken Zsh. zw. hohem BMI und Diabetes (ausgenommen einiger Ausreisser mit sehr hohem BMI und keinem Diabetes). + metrisches Merkmal (zu viele binäre, kategoriale Merkmale im Datensatz)
GenHlth (allg. Gesundheit)	<ul style="list-style-type: none"> + Intuitiv plausibel + Ordinal (immerhin 5 Kategorien) 0 Etwas unspezifisch" - mögliche semantische Überlappung mit <i>PhysHlth</i>, <i>PhysActivity</i>
PhysHlth (körperl. Gesundheit)	<ul style="list-style-type: none"> + Balkendiagramm zeigt brauchbare Trennung + Ordinal (immerhin 30 Werte) 0 zieht nur einen kurzen Zeitraum in Betracht (letzte 30 Tage) - mögliche semantische Überlappung mit <i>GenHlth</i> u. <i>PhysActivity</i>
Age	<ul style="list-style-type: none"> + Wissenschaftlich relevant (hohes Alter = Risikofaktor) + Verteilung zeigt klaren Trend + Ordinal
Income	<ul style="list-style-type: none"> + Sozialfaktor, kann auf Lebensstil (ernährung, Zugang zu med. Versorgung, Stress) hinweisen 0 Diagramm: kein starker Trend, weniger diskriminativ - Eventuell indirekter Zusammenhang
HighBP (hoher Blutdruck)	<ul style="list-style-type: none"> + Mittel-hoher wissenschaftlicher Risikofaktor + Klarer Unterschied in Verteilung
HighChol (hoher Cholesterinwert)	<ul style="list-style-type: none"> + Verwandt mit Dyslipidämie (wiss. Faktor) + Sichtbare Trennung in Verteilung
Stroke	<ul style="list-style-type: none"> + Klinisch plausibel: Schlaganfall ist häufige Komorbidität bei Diabetes, da beide Gefäßerkrankungen fördern. + Verteilung zeigt klaren Unterschied zwischen Diabetes- und Nicht-Diabetes-Gruppen

HeartDiseaseOrAttack	+ Klinisch plausibel (Komorbidität) + Starke Korrelation mit Diabetes erkennbar
PhysActivity	+ starker wiss. Risikofaktor (Körperliche Inaktivität) + Verteilung zeigt klaren Zusammenhang 0 mögliche semantische Überlappung mit <i>GenHlth</i> , <i>PhysHlth</i>
DiffWalk (Schwierigkeiten beim Gehen)	+ verweist auf "körperliche Inaktivität" (= wissenschaftl. rel. Faktor) / intuitiv plausibel als Folge oder Indikator + Verteilung: Starke Trennung erkennbar

Tab. 3.e.2: Merkmale, die nicht weiterverwendet werden

Merkmal	Begründung
Education	0 Verteilung zeigt keinen klaren Zusammenhang - korreliert mit Income: Korrelation intuitiv plausibel und ähnliche Verteilung
CholChek (fand eine ärztl. Untersuchung bzgl Cholesterin statt?)	0 Kein direkter Risikofaktor / wissenschaftl. belegter Faktor - Bedeutung unklar; möglicherweise nur von Personen beantwortet, bei denen <i>HighChol</i> = 1.
Smoker	0 kein wissenschaftl. belegter Faktor - Verteilung zeigt nur schwachen Zusammenhang
Fruits (häufiger Verzehr von Früchten)	0 Ernährung ist wissenschaftl. belegter Faktor und Früchte prinzipiell gesund; doch enthalten viele Früchte viel Fruktose 0 kein Garant für insgesamt gesunde Ernährung - Verteilung zeigt nur schwachen Zusammenhang
Veggies	0 Ernährung ist wissenschaftl. belegter Faktor und Gemüse ist sehr gesund; 0 kein Garant für insgesamt gesunde Ernährung - Verteilung zeigt nur schwachen Zusammenhang
HvyAlcoholConsump	0 Verteilung: Nur bei starken Trinkern starker Zusammenhang - derGesamtanteil an starken Trinkern ist wahrscheinlich eher klein - Korrelation mit <i>GenHlth</i> oder <i>PhysHlth</i> wahrscheinlich
AnyHealthcare (ist die Person krankenversichert?)	- kein direkter (wissenschaftl.) relevanter Zsh.; falls indirekter Zsh., dann Korrelation mit <i>Income</i> wahrscheinlich (Sozioökon. Faktor) - Verteilung zeigt nur schwachen Zusammenhang
NoDocbcCost (Arztbesuch aus Kostengründen in den letzten 12 Monaten vermieden)	- Schwacher, einseitiger Zusammenhang: nur bei NoDocbcCost = 1 erhöhter Diabetesanteil; keine Unterscheidung bei = 0 - Korreliert wahrscheinlich mit anderen sozioökonomischen Faktoren wie <i>Income</i>
Sex	- Verteilung zeigt keine starke Trennung der Gruppen

Die folgenden Balkendiagramme dienen dazu, einen groben Eindruck der Verteilungen zu erlangen. Aus Platzgründen sind diese leider nicht gut lesbar und sollten bei Interesse besser im Notebook *“projektarbeit_diabetes_final.ipynb”* angesehen werden.



2) Korrelationen der Merkmalsspalten untereinander

Um Redundanzen zu vermeiden und multikollineare Effekte zu identifizieren, wäre es sinnvoll, die Merkmale sowohl deskriptiv als auch anhand geeigneter Korrelationsmaße – wie dem Phi-Koeffizienten oder Cramer's V – näher zu untersuchen. Starke Korrelationen zwischen Prädiktoren können die Anwendbarkeit bestimmter Algorithmen beeinträchtigen. Insbesondere die Bewertung individueller Einflussgrößen kann dadurch verzerrt oder überschätzt werden, sodass deren tatsächliche Bedeutung für die Vorhersage von Diabetes nicht mehr zuverlässig interpretierbar ist. Zudem kann es zu Overfitting und einer verminderten Generalisierbarkeit der Modelle kommen – insbesondere bei der logistischen Regression. Beim Naive-Bayes-Klassifikator wird ausserdem die zentrale Annahme der Unabhängigkeit der Merkmale verletzt, was zu erheblichen Performanceeinbußen führen kann. Entscheidungsbäume gelten zwar als relativ robust, tendieren aber dazu, korrelierte Merkmale zu bevorzugen. SVMs mit nicht-linearem Kernel und Random Forests zeigen sich hingegen weitgehend unempfindlich gegenüber Korrelationen zwischen den Eingangsvariablen.

Aufgrund der kurzfristigen Projektabgabe wird in dieser Arbeit auf eine weiterführende deskriptive oder explorative Analyse verzichtet. Stattdessen erfolgt die Merkmalsauswahl auf Basis theoretischer Überlegungen und visueller Beobachtungen, wobei gezielt versucht wird, starke Abhängigkeiten zwischen den Prädiktoren zu vermeiden.

Zusammenfassend lässt sich sagen, dass Korrelationen zwischen Merkmalen in medizinischen Datensätzen besonders typisch sind, da viele Gesundheitsfaktoren biologisch, verhaltensbezogen oder sozial miteinander verknüpft sind und sich wechselseitig beeinflussen:

- *HvyAlcoholConsump* korreliert wahrscheinlich mit *GenHlth* und *PhysHlth*, da starker Alkoholkonsum häufig mit schlechter Gesundheit einhergeht.
- *Income* dürfte mit *Education*, *AnyHealthcare*, *NoDocbcCost* und ggf. *PhysActivity* korrelieren, da diese Variablen sozial-ökonomische Rahmenbedingungen und deren Auswirkungen auf Lebensstil und Gesundheitsverhalten abbilden.
- *HeartDiseaseOrAttack*, *Stroke*, *HighBP* und *HighChol* sind medizinisch miteinander verbunden (kardiovaskuläre Komorbiditäten) und könnten daher ebenfalls teilweise korrelieren.
- Einige nicht weiterverwendete Merkmale (z. B. *Smoker*, *Fruits*, *Veggies*) zeigen kaum visuelle Trennschärfe und könnten inhaltlich mit anderen Lebensstilmerkmalen überlappen, ohne zusätzlichen Informationsgewinn zu bieten.

Es bleibt noch anzumerken, dass uns durchaus bewusst ist, dass in einem engeren Sinn einige der Merkmale (wie z.B. Schwierigkeiten beim Gehen)

keine Prädiktoren sondern korrelierende (postdiktive) Merkmale von bestehendem Diabetes sind. Da es unserem Auftraggeber aber auch darum geht, bisher unerkannte Fälle von Diabetes aufzudecken und nicht etwa nur Prädiabetes festzustellen, haben wir diese Merkmale beibehalten.

4. Auswahl der Machine Learning Verfahren

Die Wahl der passenden Machine Learning Algorithmen wird unter anderem erheblich durch die Struktur der Daten beeinflusst:

- vornehmlich binäre sowie ordinale und metrische Daten
- binäre Zielvariable (Diabetes- ja/nein)
- künstlich balancierte Daten
- potenziell abhängige und untereinander korrelierte Features
- große Datenmenge mit fast 70.000 Beobachtungen
- bis auf das Alter (Mittlere Cardinality: 13 Kategorien) haben alle anderen kategorialen Features eine geringe Cardinality mit weniger als sieben Kategorien
- keine Klassenüberlappungen ("Overlapping" erkennbar)

Für viele Features wäre die Verfügbarkeit der zugrundeliegenden metrischen Rohdaten wünschenswert, weil durch ihre Kategorisierung viele Informationen und Signalstärke verloren gegangen sind. Der Datensatz hat jedoch ausreichend Beobachtungen für komplexere Modelle, keine Einschränkungen hinsichtlich One-Hot-Encoding (geringe Cardinality), und kein Bedarf an Weighting (weil er gebalanced ist). Somit erlaubt er eine große Modellfreiheit

Modellauswahl zur Vorhersage von Diabetes

Für die Vorhersage von Diabetes auf Basis eines strukturierten, überwiegend binär-kategorialen Datensatzes mit fast 70.000 (Kaggle "Diabetes Balanced") fokussieren uns auf Modelle, die gut mit kategorialen und binären Daten umgehen können, gleichzeitig robuste Resultate auf großen Datenmengen liefern und möglichst für medizinische Kontexte transparent oder zumindest nachvollziehbar bleiben. Gute Interpretierbarkeit gilt allerdings nicht bei allen ausgewählten Algorithmen.

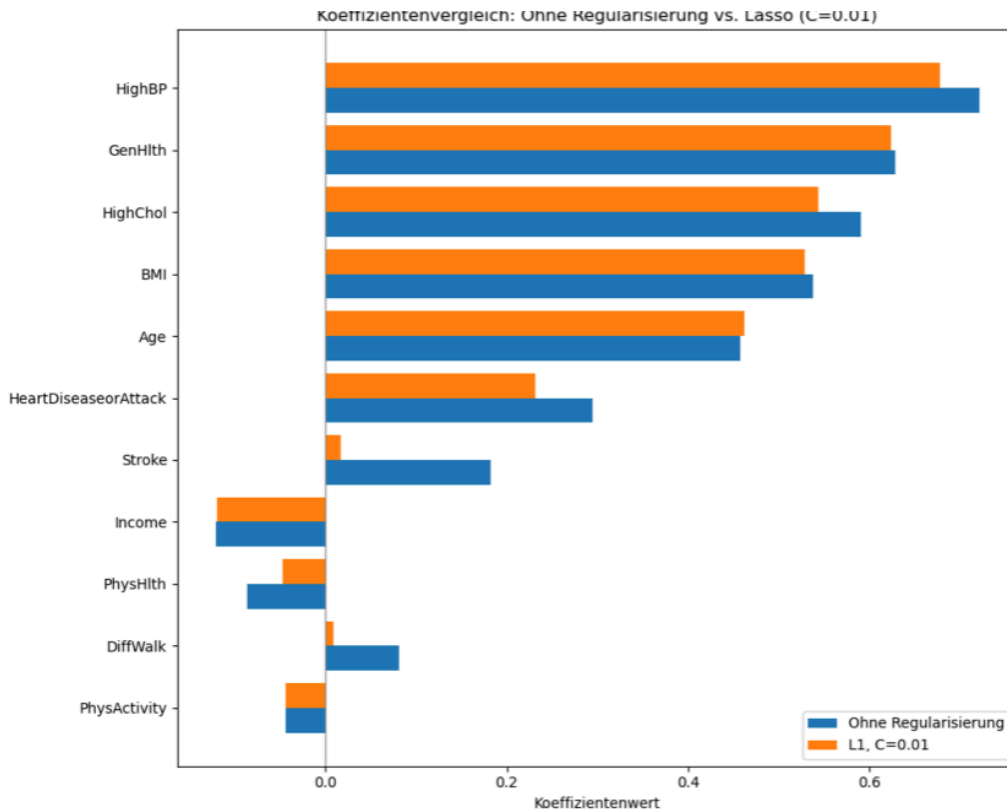
Verwendet wurden: Logistische Regression (gute Nachvollziehbarkeit und schnell), Naive Bayes (gute Nachvollziehbarkeit und schnell), Random Forest (robust und rechenintensiv) und Support Vector Machines (robust und rechenintensiv).

Bewusst ausgeschlossen wurden: K-nearest Neighbors (wenig geeignet bei vielen binär kategorialen Merkmalen und möglicher Überlappung von Merkmalen) und neuronale Netze (ebenfalls schlecht bei vielen binären Merkmalen, insgesamt zu wenige Merkmale).

5. Logistische Regression

- Motivation zum Einsatz einer logistischen Regression
 - i. Die logistische Regression ist aufgrund der binären Zielgröße, der metrischen, ordinalen und binären Eingangsvariablen sowie der guten Interpretierbarkeit eine sinnvolle Wahl für unseren Datensatz. Sie liefert transparente Einflussanalysen einzelner Risikofaktoren und ermöglicht eine ärztlich nachvollziehbare Kommunikation der Ergebnisse.
- Zentrale Parameter
 - i. Der Parameter λ definiert die Art der Regularisierung: Mit λ_2 (Ridge) werden große Koeffizienten durch einen quadratischen Strafterm geglättet, was Overfitting reduziert und bei multikollinaren Daten Stabilität schafft. λ_1 (Lasso) hingegen setzt potenziell irrelevante Merkmale auf Null – hilfreich für Feature Selection.
 - ii. Der Parameter C reguliert die Stärke dieser Bestrafung ($C = 1/\lambda$): Kleinere Werte bedeuten stärkere Regularisierung.
 - iii. Solver legt das numerische Optimierungsverfahren fest. Je nach z.B. Datensatzgröße und ob und welche Art der Regularisierung gewünscht ist, bieten sich unterschiedliche Verfahren an. Die Wahl des Solvers sollte aber, außer bei Konvergenzproblemen, die Schätzergebnisse nicht beeinflussen.
 - iv. Da das Training iterativ erfolgt, muss mit `max_iter` die maximale Anzahl an Trainingsschritten festgelegt werden; bei vielen Merkmalen oder starker Regularisierung empfiehlt sich ein erhöhter Wert (z. B. 500–1000), um Konvergenz sicherzustellen.
- Erwartetes Verhalten und Parametrisierung
 - i. Solide Vorhersagegenauigkeit. Die logistische Regression funktioniert bei der Art des Featuremixes relativ gut. Aber sie erkennt keine Featureinteraktionen und keine nichtlinearen Effekte. Dementsprechend wird erwartet, dass wir robuste Ergebnisse erhalten aber Modelle wie Random Forest bessere Vorhersagewerte liefern.
 - ii. Aufgrund der geringen Featureanzahl und der für eine logistische Regression hohen Beobachtungsanzahl erwarten wir, dass es sich sehr stabil generalisieren lässt, auch ohne Ridge Regularisierung.
 - iii. Da aufgrund der Natur unserer Daten, trotz sorgfältiger Featureauswahl, zumindest zum Teil moderate Korrelationen zwischen unseren Features zu erwarten sind, verwenden wir trotzdem auch die Ridge Regularisierung. Wir erwarten jedoch aufgrund von ii. keine deutlichen Auswirkungen.
 - iv. Da unser Modell nicht viele Features hat, erwarten wir keine große Auswirkung von der Lasso Regularisierung auf die Modell-Scores. Wir werden sie lediglich als Kontrolle anwenden und um mehr über die potenzielle Relevanz einzelner Einflussfaktoren/Features zu erfahren und um eventuelle Streichkandidaten unter diesen zu identifizieren.

- Test der Feature Importance



- Ziel: Beurteilung der Feature Importance und den Einfluss der Lasso Regularisierung.
- Eine logistische Regression ohne Lasso Regularisierung und drei logistische Regression mit Lasso Regularisierung mit Cs von 1, 0.01, und 0.001 werden miteinander verglichen. (siehe Jupyter Notebook für Auflistung der Ergebnisse und Parametrisierung)
- Im Vergleich von logistischer Regression ohne Lasso Regularisierung zu einer Lasso Regularisierung mit C=1.0 sind Änderungen in den Koeffizienten kaum bemerkbar. Für Koeffizienten finden Änderungen ab der vierten (bzw. für einen Koeffizienten dritten) Nachkommastelle statt.
- Der Vergleich mit einer Lasso Regularisierung mit einem C-Wert von 0.01 (siehe oben) zeigt starke Änderungen für Stroke (-87.8%), PhysHlth (61.6%) und DiffWalk (-77.4%). Dies ist ein Indiz, dass diese Features nicht sonderlich robust sind und keinen hohen Erklärungswert haben. In hypothetischen weiteren Feature Eliminierungszyklen wären dies Streichkandidaten. Gleichzeitig ist der Absolutwert dieser Features gering.
- HighCol (-6.4%), HighBp (-3.8%, *Hypothese H2*), und BMI (-2.4%, *Hypothese H1*) hingegen haben nicht nur mit die höchsten Absolutwerte, sondern zeigen auch die geringsten Veränderungen. Diese Werte und ggf. erscheinen besonders robust und relevant für das Modell.

- vi. Dass der Effekt von Income stark steigt, wird mit Korrelationseffekten mit den an Einfluss verlierenden Variablen zusammenhängen. Diese Änderung steht nicht für Robustheit.
- vii. PhysicalHealth und BMI als metrische Variablen wurden standardskaliert.
- viii. Income, Age und GeneralHealth sind ordinaldaten die durch das Modell wie metrische Daten behandelt werden. Bei Age sind die Abstände zwischen den einzelnen Kategorien fast gleich, weshalb solch eine Handhabung Sinn ergibt. Age hätte auch standardskaliert werden sollen. Bei Income variieren die Kategoriegrenzen stark, weshalb hier ggf. one-hot-codierung oder Kulmutlatives Binärisches Encoding sinnvoll wären. Für GeneralHealth gilt das gleiche, da die fünf Kategorien auf rein subjektiven Einschätzungen basieren und die Abstände nicht interpretierbar sind.
- ix. Leider wurde viii) von mir übersehen, so dass sich dies nicht mehr korrigieren ließ. Zum Test wurden die Koeffizienten einer logistischen Regression bestimmt, bei denen auch Income, Age und GeneralHealth standardskaliert waren. Dabei zeigte sich, dass Age deutlich an Bedeutung verlor, während GeneralHealth weiterhin einen dominanten Einfluss aufwies. Der Einfluss von Income und GeneralHealth nahm durch die Skalierung nur leicht ab. Alle übrigen Koeffizienten wurden durch diese Änderungen nur geringfügig beeinflusst.

○ Ergebnisse

	Initiales "Baseline" Model	"Bestes Modell "(L2: "Ridge")	Bestes "L1: Lasso" Modell
Parametrisierung	<ul style="list-style-type: none"> • C=1.0 • max_iter=1000 • penalty='l2' • solver='newton-cg' 	<ul style="list-style-type: none"> • C=: 0.01 • max_iter=500, • penalty='l2', • solver='liblinear' 	<ul style="list-style-type: none"> • C=0.01, • max_iter=500, • penalty='l1' • solver='liblinear'
Recall	0.7719	0.7719	0.7719
Precision	0.7461	0.7466	0.7461
Accuracy	0.7481	0.7484	0.7481
Ursprung	Ausgangsparametrisierung gemäß Literatur	Gridsearch mit Begrenzung auf l2 Regularisierung	Gridsearch mit Begrenzung auf l1 Regularisierung

- i. Aufgrund der oben stehenden Überlegung wurde der Fokus auf die l2 Regularisierung beim Gridsearch gesetzt.
- ii. Zur Kontrolle wurden weitere Gridsearch-Läufe mit Beschränkungen auf l1 Regularisierung und mit Fokus auf höhere Iterationsanzahlen durchgeführt. Außerdem wurden die obigen Scores mit einer Regression ohne Regularisierung verglichen

- iii. Die unterschiedlichen Modelle weisen vereinzelt Unterschiede in der vierten Nachkommastelle der Scores (ermittelt mit Testdaten) auf. Das "Beste Modell" mit L2-Ridge Regularisierung weist die besten Werte auf. Auch ohne Signifikanztests ist davon auszugehen, dass die Unterschiede nicht relevant sind.
- iv. Die fehlenden Unterschiede in den Scores bestätigen die oben geäußerte Vermutung, dass die L2 und L1 Regularisierung im Kontext unserer Daten nicht für die Vorhersageergebnisse relevant sind.
- v. Dass die Modelle mit L1, L2 und ohne Regularisierung die gleichen Scores liefern, ist Hinweis darauf, dass das Modell nicht overfitted, die relevanten Features deutlich zur Vorhersage beitragen, und das Modell allgemein robust ist.
- vi. Dass die Scores je Modell bei der Kreuzvalidierung um 0.01 bis 0.02 Punkte variieren, ist ein Indiz, dass die Modelle robust sind, aber doch auf Variation in den Datensätzen reagieren
- vii. Die Beurteilung der Robustheit des Modells hat eine deutliche Einschränkung: die Testdaten wurden aus einem balancierten Trainingsdatensatz gewonnen, womit die Testdaten nicht das Verhältnis der Grundgesamtheit widerspiegeln. Somit verlieren die Ergebnisse auf Basis der Testdaten an Aussagekraft.
- viii. Dass die L1 Regularisierung die gleichen Scores liefert und manche Merkmale de facto auf null setzt (z.B. DiffWalk, Stroke, PhysActivity) ist ein deutliches Zeichen dafür, dass diese Merkmale aus dem Modell entfernt werden können, die Vorhersagequalität zu gefährden.
- ix. Die logistische Regression hat wertvolle Einsichten in die Merkmalrelevanz gegeben, aber nicht die gewünschten Zielwerte für den Recall-Score erreicht.
- x. Die logistische Regression hätte vorab auf alle Datensatzdaten angewandt werden sollen, um die initiale Featureselektion zu unterstützen.

6. Naive-Bayes-Modelle

Ich habe mich auf die Naive-Bayes-Modelle konzentriert, die anhand der Features voraussagen, zu welcher Zielklasse Datensätze am wahrscheinlichsten gehört - in diesem Falls Diabetes 1 oder 0.

Zunächst habe ich zur Orientierung alle Modelle mit einem unvorbereiteten `X_train` und Default-Einstellungen trainiert, um ein Gefühl dafür zu bekommen, in welchem Rahmen sich die Scores und Laufzeit bewegen.

Die Laufzeit ist bis zum Ende unproblematisch geblieben: Die Ergebnisse konnten innerhalb weniger Sekunden berechnet und ausgegeben werden. Ziel war danach, die Scores für jedes Modell zu verbessern.

	GaussianNB	BernoulliNB	CategoricalNB
Trainingsscore	0.6939	0.7023	0.7299
CV Recall mean	0.6129	0.7118	0.7288
Precision	0.7389	0.7041	0.7357
Recall	0.6129	0.7118	0.7294
F1	0.6700	0.7079	0.7325

Am besten performt zunächst `CategoricalNB` mit einem Recall von 0.7294, gefolgt von `BernoulliNB` mit 0.7118 und `GaussianNB` zum Schluss mit nur 0.6129.

`GaussianNB` ist aber überraschend gut in der Precision und liegt dort ganz vorne mit einem Score von 0.7389.

Die Einstellung der Parameter schien nicht besonders aussichtsreich, da man beispielsweise die Glättung (falls die Wahrscheinlichkeit sehr gering oder 0 wird) und `class_prior` variieren konnte, wobei sich Letzteres auf die Verteilung der Zielvariable bezieht. Da die Zielspalte unseres Datensatzes ausgewogen ist, habe ich mich zunächst auf das auf jedes Modell individuell ausgerichtete Data Preprocessing konzentriert. Im `GridSearch` sollen dann `alpha` und `class_prior` zur Variation des Verhältnisses der Zielvariable untersucht.

Die CV-Recall-Scores sind überall aufgeführt, haben aber in Bezug auf den normalen Recall-Score nicht viel Informationszugewinn gebracht und werden daher weitestgehend ignoriert.

6.1 GaussianNB

Dieses Modell ist auf metrische, normalverteilte Daten und eine kategorische Zielspalte (liegt vor) ausgelegt. Ich habe die drei Spalten geplottet, von denen ich mir am ehesten eine Normalverteilung versprochen habe, um deren Verteilung anzusehen: BMI, Alter, Physical Health. Davon wiesen nur BMI und Alter eine normalverteilte Kurve auf. Die meisten der restlichen Spalten sind binär. Daher ist der Datensatz recht ungeeignet für dieses Modell aufgrund der wenigen metrischen Daten.

6.1.1 Datenvorbereitung

Zunächst habe ich einen Datensatz vorbereitet, indem ich die Spalte Age in einer Pipeline mit einer selbst geschriebenen Klasse neu kodiert und skaliert habe. Diese hat die Werte 1-13 mit dem Durchschnittsalter der ihnen zugeordneten Altersspanne ersetzt. Zusätzlich wurden in dieser Pipeline die beiden anderen metrischen Spalten BMI und PhysHlth skaliert (`X_train_pipe`).

Einen weiteren Datensatz, der nur die Spalten BMI und PhysHlth skaliert, habe ich ebenfalls vorbereitet. Ich war mir unsicher, ob das zuvor durchgeführt Encoding des Alters einen Nutzen hat. Da ich es bei den ordinalen Werten belassen habe, habe ich die Spalte nicht skaliert. Der Plot zeigt immerhin auch ein sehr ähnliches Bild der Verteilung der Werte wie bei den kodierten Werten (`X_train_scale`).

Zu guter letzt habe ich entschieden, den Datensatz so zuzuschneiden, dass er alle nicht-binären Spalten enthält. Somit wurden alle bis auf BMI, PhyHealth, GenHlth, Age, Income entfernt (`X_train_nonbin`).

Ein zuvor skaliertes PCA-transformierter Datensatz `X_train_pca` soll ebenfalls an dem Modell getestet werden.

6.1.2 Scores

	X_train	X_train_pipe	X_train_scale	X_train_nonbin	X_train_pca
recall	0.6129	0.6120	0.6129	0.6274	0.6769
recall CV	0.6129	0.6121	0.6129	0.6271	0.6769
train_score	0.6939	0.6935	0.6939	0.6941	0.7114

GaussianNB performt mit PCA-Datensatz mit 5 Spalten in Recall als auch Trainingsscore mit Abstand am besten. Die übrigen Transformationen haben das Ergebnis im Vergleich zum Originaldatensatz nicht nennenswert verbessert. Etwas besser war jedoch der Datensatz, der keine binären Werte beinhaltet. Dieser niedrige Score erfüllt die Erwartungen an diese Modell mit der Datenlage.

6.2 BernoulliNB

Dieser Algorithmus scheint perfekt für diesen Datensatz zu sein, da er binäre Features benötigt, die hier in einer Vielzahl vorliegen, um die binäre Zielspalte vorauszusagen.

6.2.1 Datenvorbereitung

Da Bernoulli am besten mit binären Werten operiert, habe ich den Datensatz mit einer eigenen Klasse so vorbereitet, dass er für Age und BMI nur True und False enthält (für übergewichtig/nicht übergewichtig und hohe Wahrscheinlichkeit für das jeweilige Alter, an Diabetes zu erkranken: ja/nein) - X_train_bin. Grenzwert und Spaltenindex werden der Klasse übergeben. Diese habe ich beim Alter auf 40 und beim BMI auf 25 angesetzt.

Außerdem habe ich einen weiteren Datensatz zugeschnitten, der nur die booleschen Werte beinhaltet und alle weiteren Spalten entfernt (X_train_bool), um zu sehen, ob der Score sich dadurch verbessert. Ich habe erwartet, dass andersartige Spalten den Score verwässern und daher verschlechtern.

6.2.2 Scores

	X_train	X_train_bin	X_train_bool	X_train_pca
recall	0.7118	0.7395	0.6996	0.7538
CV recall	0.7118	0.7396	0.7054	0.7538
train_score	0.7023	0.7118	0.7026	0.6954

Den besten Recall-Score hat wieder der PCA-Datensatz mit 0.7538, der aber sogar um 0.9 besser ist als der des vorherigen Modells. Den besten Trainingsscore hat X_train_bin mit der binarisierten BMI- und Altersspalte. Der Datensatz, der nur die booleschen Spalten enthält, ist am schlechtesten - sogar noch schlechter als der unvorbereitete Datensatz X_train. Offenbar kann das Modell doch den Zugewinn an Informationen von nicht-booleschen Spalten für seine Voraussagen gebrauchen.

6.3 CategoricalNB

Dieses Modell arbeitet am besten mit kategorialen Werten, weshalb ich mir hier das beste Modell erwarte: viele binäre Spalten, der Rest der Spalten ist ordinal bis auf die BMI-Spalte.

6.3.1 Datenvorbereitung

CategoricalNB arbeitet am besten mit ordinalen Daten. Deshalb habe ich pro forma den OrdinalEncoder verwendet, der jedoch keinen Informationszugewinn bedeutet (X_train_enc).

Zudem habe ich einen weiteren Datensatz vorbereitet, der die binarisierte BMI-Spalte, aber die Original-Age-Spalte enthält, da diese bereits ordinal kodiert ist (X_train_bin_bmi).

6.3.2 Scores

Die Scores für X_train und X_enc sind erwartungsgemäß gleich - es liegen ja bereits überwiegend binäre bzw. ordinale Daten im gesamten Datensatz vor. Den PCA-Datensatz habe ich hier nicht verwendet, da dieser negative Werte beinhaltet, die der CategoricalNB nicht berechnen kann.

	X_train	X_train_enc	X_train_bin_bmi
recall	0.7294	0.7294	0.7239
CV recall mean	0.7288	0.7288	0.7245
train_score	0.7299	0.7299	0.7244

Der Recall liegt hier bei 0.7294 und ist damit schlechter als der des Bernoulli-Modells, aber sehr viel besser als des Gaussian-Modells. Dieser Score ist sehr leicht zu erreichen gewesen in Anbetracht dessen, dass das Data-Preprocessing auch hätte weggelassen werden können (siehe Scores X_train).

Das Modell mit der binarisierten BMI-Spalte ist minimal schlechter, was mich etwas verwundert, da das Modell auf ordinale Daten zugeschnitten ist. Offenbar ist die Informationsverdichtung hier nicht sachdienlich gewesen.

Eine Analyse mit dem PCA-Datensatz wäre sicher vielversprechend gewesen. Diese konnte aber aufgrund von Berechnungsproblemen des Modells mit Negativwerten nicht durchgeführt werden. Hier hätte ich mich näher mit den Gründen dafür und einer anderen Transformationsmöglichkeit beschäftigen können.

6.4 GridSearch

Wie eingangs erwähnt habe ich mir durch die Variation der Parameter nicht viel versprochen, da der Datensatz bereits perfekt ausgeglichen war. Dennoch habe ich einen GridSearch mit 'alpha': [0.01, 0.1, 0.5, 1.0, 5.0] und 'class_prior': [[0.8, 0.2], [0.7, 0.3], [0.9, 1.0], None] durchgeführt.

Der Defaultwert für alpha ist 1.0, die Standardglättung. Für class_prior ist der Defaultwert None, den ich in meinen GridSearch aufgenommen habe - zusätzlich zu einer 90/10-, 80/20- und 70/30-Verteilung der Ausprägungen der Zielvariable, da Diabetes=True im Datensatz überrepräsentiert ist.

Mein bis dahin bestes Modell BernoulliNB mit einem PCA-preprocessed X_train und Defaultparametern war Ausgangspunkt meines GridSearchs. Zusätzlich habe ich aber auch noch ein GridSearch mit dem CategoricalNB-Modell mit originalem X_train und Defaultparametern ausprobiert.

Für beide Modelle hat der Gridsearch alpha=0.01, was einer sehr schwachen Glättung entspricht, und class_prior=None ergeben. Der Recall-Score war mit dem Bernoulli-Modell bei 0.7449, beim Categorical-Modell bei 0.7289.

Dieses Ergebnis ist verwunderlich, da zuvor bereits ein Modell unter Default-Parametern mit einem besseren Recall-Score gefunden wurde und in dem GridSearch diese auch berücksichtigt wurden (alpha=1.0, class_prior=None). Ich sehe daher nicht das ermittelte Modell als mein bestes an und werde im Voting das Default-Bernoulli-Modell beisteuern.

6.5 Anwendung auf Testdaten

Im letzten Schritt wurden pro Modell die am besten performenden Datensätze mit dem entsprechenden X_test getestet: GaussianNB mit X_train_pca, Bernoulli mit X_train_pca, aber auch X_train_bin, da im Voting keine weitere PCA mit 5 Spalten trainiert wurde, und CategoricalNB mit X_train. Bernoulli mit X_train_bin sollt später für das Voting verwendet werden.

	Gauss + X_test_pca	Bernou. + X_test_pca	Bernoulli + X_test_bin	Categorical + X_test
recall	0.6836	0.7449	0.7376	0.7286
CV recall mean	0.6954	0.7998	0.7348	0.7288
accuracy	0.7200	0.7031	0.7205	0.7379

Auf den Testdaten ist wieder Bernoulli mit den PCA-Daten am besten, gefolgt von Bernoulli mit den binär umgewandelten Daten für Age und BMI.

Abschließend lässt sich sagen, dass, wie geahnt, das Gaussian-Modell nicht besonders gut passt. Das Bernoulli-Modell stellt sich als am besten heraus: Es bringt die besten Scores mit Training mit dem PCA-Datensatz und dem X_train_bin-Datensatz, die beide noch vor den Scores des CategoricalNB-Modells liegen. Dass das Bernoulli auf diesen Daten arbeitet verwundert mich. Dies liegt offenbar auch nicht nur an den PCA-vorbereiteten Daten, da hier auch das Modell mit den binären Age- und BMI-Spalten besser war als das CategoricalNB-Modell.

6.6 Bestes Modell auf den unbalancierten dritten Datensatz anwenden

Das Bernoulli-Modell, trainiert mit PCA-vorbereiteten Daten, wurde zum Schluss auf den unbalancierten dritten Datensatz angewendet, der eine Verteilung von 86% (Diabetes 0) zu 14% (Diabetes 1) aufweist. Das Modell hat auf diesen Daten (ebenfalls mit PCA) einen Recall von 0.8214, was erstaunlich ist, da der Recall für die Testdaten nur 0.7449 betrug. Mit etwas mehr Zeit wäre es sehr spannend gewesen, herauszufinden, wie dieses im Vergleich zum Trainings- und Testscore sehr gute Ergebnis zustande kommt und wie die anderen Modelle ihre Vorhersagen getroffen hätten.

7. Random Forest

- Motivation zum Einsatz von Random Forest
 - i. Random Forest ist gegenüber Decision Tree zu bevorzugen, da der Algorithmus als robuster gilt und weniger zu Overfitting neigt
 - ii. Random Forest kann mit den vorliegenden Daten gut umgehen. Eine Reduktion von 22 auf 11 wäre nicht zwingend notwendig gewesen. Mit dem Verfahren hätte man die Feature Importance auch ex post untersuchen können. Es ist keine Skalierung oder weitere Datenvorbereitung notwendig. Wir verwenden die Random Forest Klassifikation (RandomForestClassifier)
 - iii. Als Nachteil ist die geringe Interpretierbarkeit zu nennen. Dies war eine Anforderung des Auftraggebers. Sowie der im Vergleich zu den anderen Verfahren höhere Rechenaufwand (dieser hält sich aber bei 70.000 Datensätzen im Rahmen).
- Erster Durchlauf mit Default-Einstellungen

Es soll zunächst eine robuste Basis gelegt werden.
Zur Bewertung der Güter werden accuracy, recall, precision und F1 Score herangezogen. Ziel Score ist laut Auftraggeber recall

Score	Trainingsdaten	Testdaten
Recall	0.9356	0.7409
Precision	0.9488	0.7143
Accuracy	0.9418	0.7149
F1 Score	0.9421	0.7273

Bewertung: Mit einem recall Score von 0.7409 auf den Testdateen sind wir nicht zufrieden. Der Vergleich zu den Testdaten zeigt starkes Overfitting des Models.

- Grid Search Runde 1
Einstellung des Parameter Rasters

Parameter	Spezifikation	Motivation
n_estimators	50,100,200	Erstmal weniger Bäume, Fokus auf Rechen-Zeit und schnelle erste Ergebnisse
max_depth	5,10,20	Begrenzte Tiefe. Wir wollen im Vergleich zum "Ersten Durchlauf" (s.o.) Overfitting vermeiden.
min_samples_split	2,5,10	Höhere Werte fördern Generalisierbarkeit
min_samples_leaf	1,2,4	Mindestanzahl von Samples in Endknoten. Analog zu 'min_samples_split', aber mit stärkerem Fokus auf Blattknoten-Stabilität

Ergebnisse Grid Search Runde 1

Score	Trainingsdaten	Testdaten
Recall	0.8109	0.7981
Precision	0.7407	0.7401
Accuracy	0.7602	0.7525
F1 Score	0.7742	0.7742

Bewertung: Der recall Score hat sich von 0.7409 im ersten Lauf zu 0.7981 auf den Testdaten verbessert. Es ist kein Overfitting mehr zu beobachten. Die Scores auf Trainings- und Testdaten haben sich angenähert

- Weitere Gridsearch-Verfahren
Es wurden noch drei weitere Gridsearch Verfahren mit unterschiedlichen Parameter-Einstellungen durchgeführt. Die Ergebnisse sind unten zu sehen. (Details im Jupyter Notebook)

Durchlauf	Fits	Recall(Test)	Beste Parameter Konfiguration
Grid_search_1	540	0.7981	'max_depth': 10, 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 200
Grid_search_2	1080	0.7981	max_depth': 10, 'max_features': 'sqrt', 'min_samples_leaf': 4, 'min_samples_split': 10, 'n_estimators': 200
Grid_search_3 ¹ SIEGER	2700	0.8000	'bootstrap': True, 'max_depth': 9, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 250
Grid_search_4 ²	250	0.7987	bootstrap': True, 'max_depth': 8, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 9, 'n_estimators': 231

- Fazit und Ausblick:

Durch das Testen verschiedener GridSearch Verfahren konnte der Recall-Wert auf den Testdaten verbessert werden – von 0.7409 im Ausgangsmodell auf bis zu 0.8000 im besten Fall. Die Zielsetzung von 0.9 wurde nicht erreicht. Eine Auswertung und Grafik zu den Feature Importances findet sich im Jupyter Notebook. In einem nächsten Verfahren könnte man auf die Vorauswahl der Variablen verzichten und dies im Rahmen des Random Forest Verfahrens ex post festlegen bzw. die Modelle mit verschiedenen Features optimieren.

¹ Im dritten Grid-Search-Verfahren wurden zusätzlich die Parameter **bootstrap** und **max_features** berücksichtigt. Mit bootstrap=True wurde das Modell auf zufälligen Stichproben mit Zurücklegen trainiert, was die Stabilität erhöht. max_features='sqrt' bedeutet, dass bei jedem Split nur ein Teil der Merkmale betrachtet wird, was die Vielfalt der Bäume fördert und Overfitting reduziert.

² Im vierten Grid-Search-Verfahren wurde mit RandomizedSearchCV gearbeitet: 50 Iterationen

8. Support Vector Machines

- Motivation zum Einsatz von Support Vector Machines
 - i. SVMs eignen sich gut für binäre Klassifikationsaufgaben - durch die Maximierung des Abstands zwischen den beiden Klassen mittels einer Hyperebene helfen Support-Vektor-Maschinen (SVMs) Overfitting zu vermeiden
 - ii. SVMs können gut mit heterogenen Datensätzen wie dem Diabetes-Datensatz umgehen, da sie sowohl mit (binär) kategorialen als auch ordinalen und metrischen Merkmale arbeiten können, solange diese numerisch sind. Es wird auch keine bestimmte Verteilung der Merkmalswerte vorausgesetzt
 - iii. Im Unterschied zu Naive Bayes sind SVMs unempfindlich gegenüber Korrelationen zwischen Merkmalen, da sie keine Unabhängigkeit voraussetzen. Auch bei redundanten Informationen kann die Methode eine stabile Trennfläche finden (Nachteil: sie werden allerdings übergewichtet, SVM bietet jedoch Möglichkeiten, um dem entgegenzuwirken (Kernelwahl; Parameter: *class_weight*, *sample_weight*).

Nachteile des Algorithmus: ggf. aufwendige Vorbereitung nötig: SVMs reagieren empfindlich auf fehlende oder schlechte Skalierung. Vor allem aber sind die Ergebnisse weniger interpretierbar als bei den übrigen Algorithmen.

- Skalierung

Die Skalierung ist für die Performanz von SVMs entscheidend, da sie verhindert, dass Merkmale mit großen Wertebereichen kleinere dominieren. Daher wurden nur nicht-binäre Merkmale skaliert. Nach Recherche fiel die Wahl auf den *MinMaxScaler*, da dieser Werte in den Bereich [0, 1] bringt und so besser mit binären Merkmalen harmoniert. Eine Standardisierung (Mittelwert 0, Std. = 1) ist für SVMs nicht zwingend nötig und kann mit polynomialem Kernel sogar nachteilig sein. Da SVMs auf Support-Vektoren fokussieren und entfernte Ausreißer ignorieren, ist die robuste Wirkung des StandardScalers hier weniger relevant. Da sich SVMs ausschließlich auf die Support-Vektoren konzentrieren und entfernte Datenpunkte ignorieren, sind sie von Natur aus robust gegenüber Ausreißern. Daher benötigen sie nicht die nivellierende Wirkung des StandardScalers, die normalerweise dazu dient, den Einfluss extremer Werte zu reduzieren.

- Berechnung der Scores unter Verwendung verschiedener Kernels und Parameter

Fast alle folgenden Trainingsläufe wurden mit der Klasse SVC durchgeführt. Die Klasse NuSVC wurde kurz angetestet. Sie nutzt statt *C* den Parameter *ν* (nu), mit dem man direkt die maximale Fehlerquote und minimale Anzahl an Support-Vektoren steuern kann (nu = 0.1 bedeutet: max. 10 % der

Trainingsdaten dürfen falsch klassifiziert werden (Fehlertoleranz); Mindestens 10 % der Trainingsdaten werden zu Support-Vektoren. Aus Zeitgründen aber kein weiterer GridSearch durchgeführt, da der Testlauf keine signifikant besseren Ergebnisse lieferte.

Tab. 8.1 Trainingsergebnisse SVM

Lau f	Bemerkung	Kern el	C	Gamm a (rbf /poly only)	Recall Train	Precisi on Train	Accura cy Train	Recall Test	Precisi on Test	Accurac y Test
1	default; unskaliert	rbf	1	“scale”	0.8030	0.7188	0.7409	0.8072	0.7338	0.7507
2	default	rbf	1	“scale”	0.7992	0.718	0.5070	0.7971	0.7117	0.7379
3	GridSearch bestes lineares Modell	linear		-	0.7791	0.7238	0.7377	-	-	-
4	GridSearch bestes rbf	rbf	1	0.1	0.8018	0.7146	0.7371	-	-	-
5	GridSearch bestes poly	poly	1	0.2	0.7886	0.7225	0.7392	-	-	-
6	NuSVC	rbf	nu= 0.2	0.1	0.6054	0.6017	0.6020	0.5951	0.5888	0.5909

- Interpretation der Scores und ggf. weitere Erläuterung der Parameter
 - SVC mit Skalierung nutzt 56 % der Trainingsdaten als Support-Vektoren, unskaliert sogar 60 %, was auf eine unsaubere Trennbarkeit durch ungleich skalierte Merkmale hinweist. NuSVC begrenzt dies explizit auf 37 %, was zwar die Komplexität reduziert, aber auch zu schlechterer Leistung führt. Ein hoher Anteil an Support-Vektoren ist beim Diabetes-Datensatz erwartbar, da viele Merkmale binär sind und die Trennlinien dadurch unscharf verlaufen – das Modell bleibt bei solchen diffusen Klassenverteilungen naturgemäß „unsicher“.
 - Der Einfluss der Skalierung fällt geringer aus als erwartet (vgl. Lauf 1 und Lauf 2. Selbst ohne Skalierung erzielt das Modell bereits relativ gute Werte – ein Overfitting ist nicht erkennbar. Eine mögliche Erklärung: Beim verwendeten RBF-Kernel hat das MinMax-Scaling nur begrenzten Effekt auf die Abstandsberechnungen, da viele Merkmale ohnehin bereits im Bereich von 0 bis 1 lagen. Nur wenige Merkmale – wie z. B. der BMI – sind metrisch und weisen größere Wertebereiche auf. Eventuell wäre der **StandardScaler** doch die geeignetere Wahl gewesen: Er zentriert die Daten um den Mittelwert und skaliert auf eine einheitliche Standardabweichung. Dadurch werden Merkmale mit hoher Varianz stärker „gedämpft“, was die

Verteilung symmetrischer macht und oft robuster gegenüber Ausreißern ist – insbesondere bei Kernel-Methoden wie SVMs mit RBF-Kernel. Entgegen der initialen Annahme sind SVMs doch nicht in allen Fällen robust gegenüber Ausreißern: Ausreisser können ebenfalls Supportvektoren sein.

- Die Defaulteinstellungen in **Lauf 2** mit Skalierung führen zu schwachen Overfitting. Möglicherweise ist hier der C-Wert zu hoch und die “Straße” somit zu breit.
- Bei “GridSearch” in Lauf 3 wurden folgende Parameter gesetzt:
 - `'kernel': ['linear'],`
`'C': [0.1, 0.2, 1, 10]`
 - `'kernel': ['rbf'],`
`'C': [0.1, 0.2, 1, 10],`
`'gamma': [0.01, 0.1, 0.2, 1]`
 - `'kernel': ['poly'],`
`'C': [0.1, 0.2, 1],` # weniger C- Werte für poly (poly ist langsamer)
`'gamma': [0.01, 0.1, 0.2],`
`'degree': [2]` # kleiner als default = 3 um Performanz zu erhöhen

Da der Defaultwert für C sich als möglicherweise zu hoch angesetzt war, wurden zwei kleinere Werte (C= 0.1 und 0.2) hinzugefügt.

- `SVC.class_weight="balanced"` Obwohl dies auf den ersten Blick nicht als nützlich für bereits ausbalancierte Daten zu sein scheint, kann es trotzdem bei der Klassifikation helfen, da es auch ungleich verteilte Support Vektoren nahe den “Rändern” der Strasse ausgleicht, so dass die Klasse mit den meisten “Randvektoren” nicht favorisiert wird.
- Für die Cross-Validation wurde eine 5-fache Aufteilung (5-Fold) gewählt. Dabei kam die Methode *StratifiedKFold* zum Einsatz, um sicherzustellen, dass die Klassenverteilung in jedem Fold der des Gesamtdatensatzes entspricht.
- Aufgrund langer Laufzeiten mussten einige Einschränkungen in Kauf genommen werden (weniger C-Werte und einen kleinen “degree” Wert beim Polynomial-Kernel (default=3), der bei stark nicht-linearen Entscheidungsgrenzen zu schwach sein kann).

Das beste GridSeach-Ergebnis (Lauf 4, kernel=rbf) hat jedoch wieder - wie beim überangepassten Ergebnis von Lauf 2 - einen relativ hohen C-Wert (C=1). Der moderate Gamma Wert (0.1) vermeidet hoffentlich sowohl Über- als auch Underfitting, indem lokale Variation in den Daten zwar berücksichtigt, aber nicht überbewertet wird. Der Recall in den einzelnen Folds variiert nicht zu stark (max ca. 0.5%), zumindest also scheinen die Ergebnisse robust zu sein (siehe Notebook).

Der lineare Kernel schnitt erwartungsgemäß am schlechtesten ab, da medizinische Zusammenhänge meist nicht linear trennbar sind. Der polynomiale Kernel blieb hinter dem RBF zurück, was teils am begrenzten Parametersuchraum lag (geringer degree-Wert). Auch wurde `coef0` nicht variiert.

Dennoch ist das Default-Modell (Lauf 1) leicht besser als das beste Grid-Search Modell, was auf eine suboptimale Wahl der Skalierung und/oder der GridSearch Parameter schließen lässt!

9. Zusammenführung und Interpretation der Ergebnisse

Modell	Recall(Test)	Precision(Test)	Accuracy(Test)
Logistische Regression	0.7719	0.7466	0.7484
Bayes (BernoulliNB mit PCA)	0.7445	0.6957	0.7017
Random Forest	0.8000	0.7398	0.7529
SVM	0.8072	0.7338	0.7507

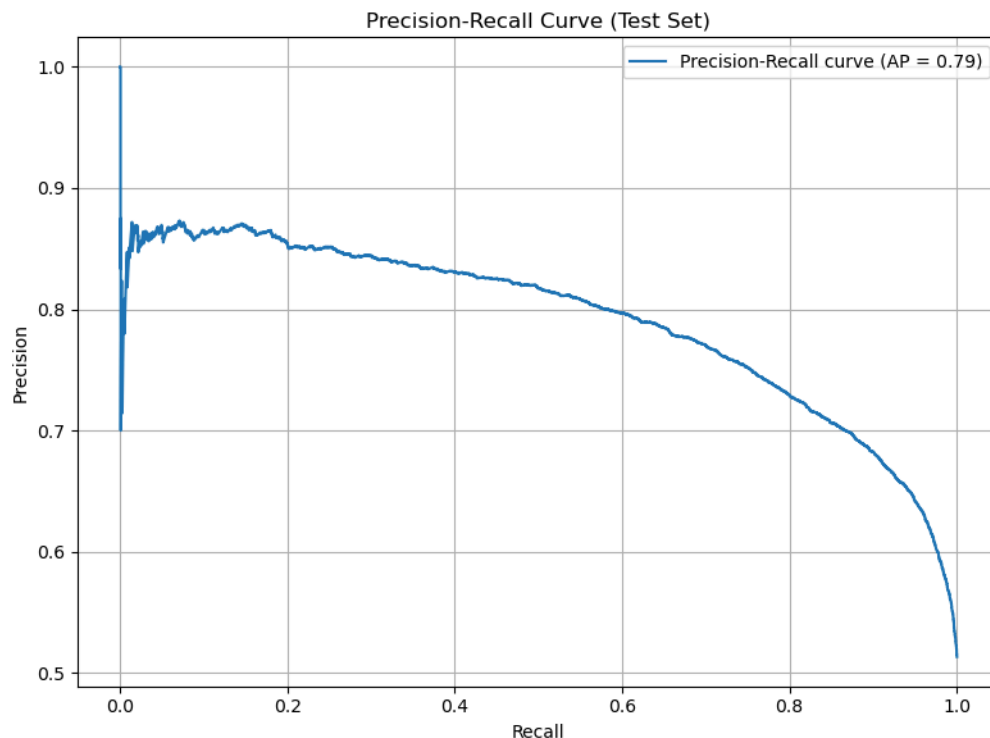
Zunächst muss festgestellt werden, dass kein Modell den geforderten Recall-Wert von 0.9 erreicht hat. Der Vergleich der vier getesteten Modelle zeigt einige Unterschiede in ihrer Leistungsfähigkeit auf den vorliegenden strukturierten, binär-kategorialen Gesundheitsdaten, auf die im weiteren kurz eingegangen werden soll:

Support Vector Machines (SVMs) und Random Forests identifizieren mit einem Recall über 0.80 und guten Precision-Werten (SVM: 0.73, RF: 0.74) am zuverlässigsten Risikofälle. Die logistische Regression liefert ausgewogene Ergebnisse mit einem Recall von 0.77 und einer Precision von 0.75, und bietet zugleich den höchsten Grad an Modelltransparenz, was sie hinsichtlich der Aufgabenstellung (hohe Nachvollziehbarkeit) attraktiv macht. Naive Bayes weist einen Recall von 0.74 auf, zeigt aber dennoch eine gute Leistung, die durch hohe Geschwindigkeit und einfache Umsetzung punkten kann, besonders in frühen Analysephasen oder bei ressourcenschwachen Anwendungen.

Für die Aufgabenstellung "Risikoerkennung von Diabetes" empfehlen wir den Random Forest, da er eine hohe Vorhersagekraft mit Stabilität und geringer Overfitting-Gefahr kombiniert – ideal für große strukturierte Datensätze. Bei Bedarf nach voller Transparenz und leichter kommunizierbaren Ergebnissen (z. B. ärztliche Beratung oder Policy-Kontext) ist die logistische Regression weiterhin eine wertvolle Option. SVMs sind ebenfalls leistungsfähig, erfordern aber höhere Rechenressourcen (Zeitkomplexität: $O(n^2)$ bis $O(n^3)$) und sind weniger intuitiv interpretierbar. Naive Bayes eignet sich primär als leichtgewichtige Benchmark oder Vorscreening-Modell.

Precision-Recall Kurven veranschaulichen visuell, ab welchem Recall-Wert die Precision sinkt:

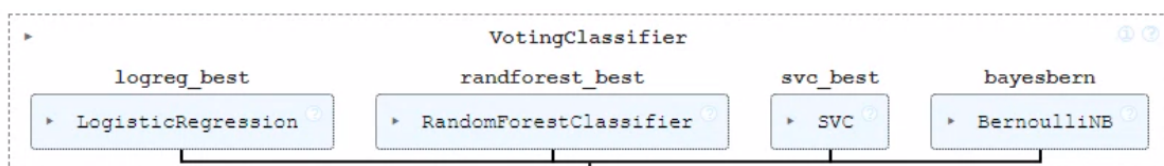
Precision-Recall Kurve für das beste Einzelmodell (SVM)



Ähnlich wie bei der “Elbow”-Methode lässt sich hier ein “Sweet Spot” feststellen, bei dem der Recall noch hoch ist und die Precision noch akzeptabel. Man sieht deutlich, wie die Precision bei Recall-Werten über 0.8 rapide absinkt.

10. Voting

Anschließend wurde das Ensembleverfahren VotingClassifier angewendet. Dabei wurden die im vorherigen Kapitel ermittelten besten Modelle (logistische Regression, BernoulliNB, Random Forest und SVM) kombiniert.



Als Datengrundlage diente die auf 11 Features reduzierte Version von X_{train} . Eine besondere Herausforderung bestand darin, dass die Modelle ursprünglich auf unterschiedlichen, individuell optimierten Datengrundlagen trainiert worden waren.

Model	Specs	Verwendete Datengrundlage der Optimierung
log REg	solver='liblinear', max_iter=500, random_state=42, penalty="l2", C=0.01	StandardScaler auf Spalten 2 und 7 angewandt
BernoulliNB	{'alpha': 0.01, 'class_prior': None}	X_train_pca: PCA mit n_components=5 + StandardScaler
Random Forest	'bootstrap': True, 'max_depth': 9, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 250	auf 11 Features reduzierter X_train
SVM	class_weight='balanced', 'C': 1, 'gamma': 0.1, 'kernel': 'rbf'	MinMaxScaler, nur nicht-binäre Features skaliert; cv folds stratifiziert

Um eine einheitliche Grundlage für das Ensembleverfahren zu schaffen, wurde als Kompromiss der StandardScaler auf die beiden numerischen Spalten *BMI* und *Physical Health* angewendet und somit das skalierte X_train für das Voting genutzt.

Dadurch konnten für Bernoulli, SVC und RandomForest nicht der vorbereitete Datensatz genommen werden. SVC wurde mit einem MinMax-skalierten, das beste Bernoulli-Modell mit einem pca-vorbereiteten und RandomForest ohne preprocessten Datensatz trainiert.

Die Konfiguration des Voting-Verfahrens erfolgte zunächst mit Default-Einstellungen, d. h. mit hard-voting und ohne Gewichtung der Einzelmodelle.

Im zuerst verwendeten Hard-Voting basiert die Entscheidung auf dem Mehrheitsvotum der Modelle, ohne Berücksichtigung der jeweiligen Vorhersagesicherheit. Im Gegensatz dazu kann Soft-Voting die Modellstimmen nach ihrer Wahrscheinlichkeit gewichten und somit zu einer besseren Gesamtvorhersage führen.

Zunächst wurden alle Modelle gleich gewichtet. Schwächere Modelle wie BernoulliNB könnten durch gleichgewichtete Beiträge die Vorhersagekraft des Modells "verwässern".

Deshalb wurden diese Parameter im Verlauf in Soft-Voting und gewichtete Modelle geändert und haben den Score verbessert (voting='soft', weights = [0.1,0.5,0.3,0.1]).

Die Cross-Validation mit 5 Folds ermittelte eine durchschnittliche Accuracy-Score von 0.7418 bei den Trainingsdaten und blieb dabei hinter den besten Einzelmodellen (z. B. Random Forest: 0.7529) zurück.

Ergebnisse Trainingsdaten

CV Recall Train	[0.7837 0.7864 0.7866 0.7765 0.7734]
CV Recall Train mean	0.7813
CV Scores Train	[0.7468 0.7429 0.7456 0.7378 0.7353]
CV Scores Train mean	0.7417

Ergebnisse Testdaten

Die Testdaten performen etwas besser. Der Recall liegt hier bei 0.7859, mit Crossvalidierung sogar bei 0.7905.

Precision	0.7445
Recall	0.7858
F1-Score	0.7646

CV Recall Test	[0.7961 0.7912 0.7919 0.7863 0.7868]
CV Recall Test mean	0.7905
CV Scores Test	[0.7546 0.7575 0.7498 0.7494 0.7447]
CV Scores Test mean	0.7512

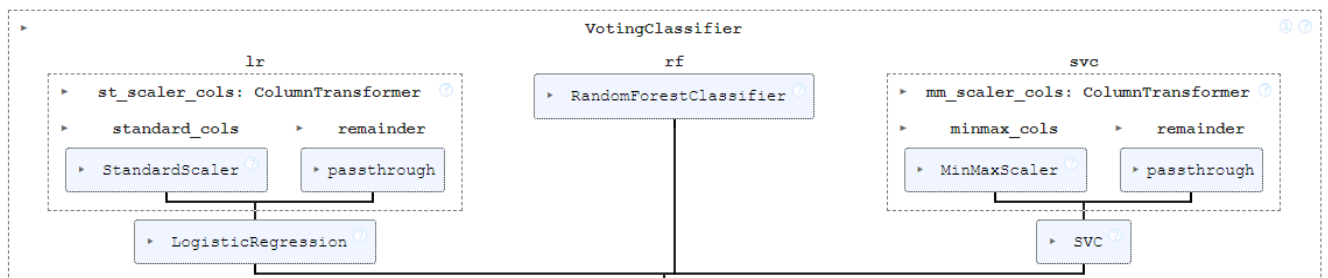
Der Rückgang der Gesamtleistung im Vergleich zu den Einzelmodellen ist kontraintuitiv, da Ensembleverfahren darauf ausgelegt sind, bessere Leistungen zu erzielen. Das Projekt-Team geht davon aus, dass die ursprünglich heterogene Datenvorbereitung ursächlich dafür sein könnten.

Die Einzelmodelle wurden ursprünglich auf speziell zugeschnittene Datenpräparationen trainiert (z. B. mit oder ohne PCA, Skalierung, Feature-Auswahl). Das Voting-Modell hingegen basiert auf einer vereinheitlichten, kompromissbasierten Vorverarbeitung. Dies kann dazu führen, dass die Modelle nicht unter ihren jeweils optimalen Bedingungen arbeiten und in ihrer individuellen Stärke eingeschränkt sind.

12. Voting-Pipeline

Mit einer Pipeline fürs Voting wurde nun versucht, den Score durch die individuelle Datentransformation für jedes Modell zu verbessern.

Für die LogisticRegression wurden die Spalten *BMI* und *PhysHlth* mit einem StandardScaler und für die Support Vector Machine alle numerischen Spalten mit dem MinMaxScaler transformiert. Für den RandomForest ist keine Datentransformation erforderlich. Für das Bernoulli-Modell wurden die Werte der Spalten BMI und Age in binäre Werte (übergewichtig/nicht übergewichtig und aufgrund des Alters für Diabetes gefährdet: ja/nein) transformiert, was aber nicht abgebildet ist, da dieses Modell für einen schlechteren Score im Voting verantwortlich war und es deshalb weggelassen wurde.



Ergebnisse Trainingsdaten

CV Recall Train	[0.7955 0.7977 0.7998 0.7877 0.7847]
CV Recall Train mean	0.7931
CV Scores Train	[0.7475 0.7434 0.7461 0.7371 0.7347]
CV Scores Train mean	0.7418

Der CV Recall-Score auf den Trainingsdaten hat sich minimal von 0.7905 auf rund 0.7931 verbessert.

Precision	0.7445870088211708
Recall	0.7858654253068134
F1-Score	0.7646695491043854

Der Recall ist mit der Pipeline von 0.7859 auf 0.7983 angestiegen, während die Precision von 0.7446 auf 0.7389 gesunken ist.

CV Recall Test	[0.8110 0.7983 0.8046 0.7983 0.7925]
CV Recall Test mean	0.8009
CV Scores Test	0.7567 0.7546 0.7458 0.7483 0.7429
CV Scores Test mean	0.7497

Die maßgeschneiderte Datenvorbereitung für jedes Modell des Votings hat nur eine minimale Verbesserung beim CV Recall erwirkt: 0.8009 vs. 0.7905. Der CV Accuracyscore ist mit der Pipeline sogar etwas schlechter geworden: 0.7497 vs. 0.7512.

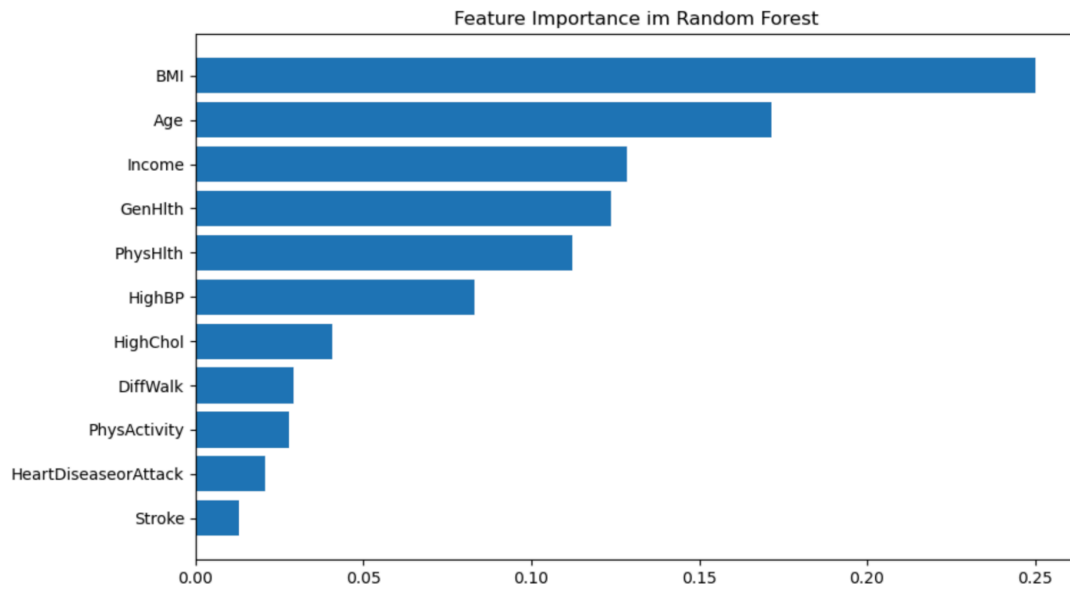
Dieser beste Score ist außerdem wahrscheinlich durch das Weglassen des Bernoulli-Modells zustande gekommen, da das den Recall-Score im Voting um 0.1 verschlechtert hat. Wir haben uns daher entschieden, es wieder herauszunehmen. Das wurde beim vorherigen Voting aus Zeitgründen nicht gemacht und ist daher nicht exakt vergleichbar mit den Voting-Pipeline-Ergebnissen.

Erstaunlich ist, dass sich der Score durch das aufwendigere individuelle Datapreprocessing nicht wesentlich verbessert hat. Hier sollte man Aufwand und Nutzen abwägen und abschätzen, ob diese Differenz für die Anwendung reale Auswirkungen hätte.

13. Schlussbemerkungen

- Das Ziel unseres Auftraggebers, ein interpretierbares Modell zur Vorhersage von Diabetes mit einem Recall von 0.90 (Hypothese H5) zu erreichen, wurde nicht erfüllt. Die besten erzielten Modelle verfehlten den Zielwert um etwa zehn Prozentpunkte.
- Ein wesentlicher Grund hierfür liegt wahrscheinlich in der Art der verfügbaren Daten: Durch die Kategorisierung ursprünglich metrischer Variablen gingen Informationen und Varianz verloren, was die Vorhersagekraft der Modelle einschränkt. Insbesondere Modelle wie SVMs oder lineare Regressionsverfahren hätten von differenzierteren, metrischen Eingabedaten profitiert.
- Trotzdem konnten mehrere Hypothesen bestätigt werden: Die Prädiktoren BMI (H1), Bluthochdruck (H2), Alter (H3) und körperliche Inaktivität (H4) zeigten in der logistischen Regression konsistente und substantielle Koeffizientenbeiträge. Vor allem BMI, Bluthochdruck und Alter erwiesen sich als prädiktiv besonders stark. Ihre Relevanz blieb auch unter Anwendung einer L1-Regularisierung (Lasso) bestehen, was ihre robuste Bedeutung im Modellkontext unterstreicht. Die Bewertung dieser Merkmale durch Fachkreise wurde durch unsere Analysen somit bestätigt. Auch der Random

Forest gewichtet zwei der drei wichtigsten wissenschaftlichen Merkmale unter seinen Top 3 Merkmalen:



- Ein weiteres zentrales Ergebnis betrifft die Robustheit der Vorhersagen. Die Testdaten wurden aus einem balancierten Trainingsdatensatz gewonnen, was bedeutet, dass das Verhältnis von Positiv- zu Negativfällen nicht dem in der realen Grundgesamtheit entspricht. Dadurch ist die Aussagekraft unserer Modelle für reale Anwendungen eingeschränkt.
- Zwar erfolgte eine Untersuchung des dritten (nicht balancierten) Datensatzes: Aus Zeitgründen wurde dieser aber nur mit dem rechenökonomischen Modell Bernoulli Naive Bayes analysiert. Dabei ergab sich ein bemerkenswerter Test-Recall von 0.8215 – deutlich besser als auf dem balancierten Datensatz (0.7445). Dieses Ergebnis hätte mit weiteren Modellen repliziert und vertieft diskutiert werden sollen. Mit mehr Zeit hätte eine vertiefte Evaluation auf einem unbalancierten Datensatz erfolgen sollen.
- Der Einsatz des Ensembleverfahrens (Voting) brachte nicht die erhoffte Leistungssteigerung. Rückblickend hätte dieses Verfahren systematischer erforscht werden können – etwa durch den Vergleich von Hard- und Soft Voting, die Variation von Modellgewichten oder alternative Datenvorbereitungsstrategien. Auch eine tiefere Auseinandersetzung mit der Frage, ob Modelle auf einer gemeinsamen oder modellspezifischen Featurebasis kombiniert werden sollten, wäre lohnenswert gewesen.
- Was wir mit mehr Zeit zusätzlich durchgeführt hätten:
 - i. Feature Selektion und Datenanalyse: Eine genauere Untersuchung der Zusammenhänge und Überlappung einzelner Merkmale (z. B. *GenHlth* vs. *PhysHlth*) hätte stattfinden können. Ob Skalierung und Zusammenfassung oder eine PCA geeigneter gewesen wären, blieb unbeantwortet. Auch ein stärker reduzierter Variablensatz hätte getestet werden können

- ii. Zusätzliche Analyseverfahren: Auch eine explorative Clusteranalyse hätte durchgeführt werden können, um zusätzliche Muster in den Daten zu erkennen.

14. Beschreibung von python-Funktionen oder Klassen, die sich gut für eine Wiederverwendbarkeit eignen

Hilfsfunktionen wie `evaluate_classifier()` sind für alle Algorithmen nützlich und können wiederverwendet werden.

```
##### Funktion zur Beurteilung von estimators
def evaluate_classifier(model, X, y, label=""):
    y_pred = model.predict(X)
    accuracy = model.score(X, y)
    precision = precision_score(y, y_pred, zero_division=0)
    recall = recall_score(y, y_pred, zero_division=0)
    f1 = f1_score(y, y_pred, zero_division=0)

    print(f"\n--- Ergebnisse für: {label} ---")
    print(f"Accuracy : {accuracy}")
    print(f"Precision: {precision}")
    print(f"Recall   : {recall}")
    print(f"F1-Score  : {f1}")
```

- Input: instanziiertes, gefittetes Modell; X-Daten; y-daten; ein label für den geprinteten Output
- Output: Accuracy-Score; Precision-Score; Recall Score; F1-Score;
- außerdem gibt die Funktion die aufgeführten Scores an die Konsole aus
- Ziel: Zeitersparnis bei der Bewertung/ dem Vergleich verschiedener ML Modelle anhand der aufgeführten Testscores