FH Vorarlberg
University of Applied Sciences

Projektarbeit Data Science

Klassifikation von Glas

Seminararbeit

Fachhochschule Vorarlberg
Energietechnik und Energiewirtschaft

Betreut von
Dr. Klaus Rheinberger

Vorgelegt von
Gstrein Michael Ulrich

Dornbirn, 30.07.2017

Inhaltsverzeichnis

Darstellungsverzeichnis

1. Einleitung

Dieser Abschnitt der Arbeit befasst sich mit der Ausgangslage, mit dem Ziel der Analyse und mit der Fragestellung.

1.1 Ausgangslage

Auf Tatorten werden häufig Glassplitter gefunden. Um Verbrechen aufzuklären ist es nützlich das gefundene Stück Glas einem Glastypen zuzuordnen und es dann als Beweisstück zu verwenden. Als Glastypen sind die verschiedenen Formen von Glas, wie sie im Alltag vorkommen zu verstehen. In diesem Fall wurden die Gläser auf folgende sieben verschiedene Herkünfte aufgeteilt:

1. Fenstergläser hergestellt mit Floatverfahren
2. Fenstergläser hergestellt mit anderen Verfahren
3. Autoscheiben hergestellt mit Floatverfahren
4. Autoscheiben hergestellt mit anderen Verfahren
5. Glasbehälter (z.B. Gurkenglas)
6. Trinkgläser
7. Scheinwerfer

Bei der Untersuchung der Gläser, welche als zukünftige Beweismittel dienen sollen, wurden die verschiedenen Inhaltsstoffe beziehungsweise die verschiedenen Eigenschaften beleuchtet. Die folgenden wurden für die Einteilung verwendet:

1. RI: Brechungsindex des Glases (refractive index)
2. Na: Natrium (sodium)
3. Mg: Magnesium (magnesium)
4. Al: Aluminium (aluminum)
5. Si: Silizium (Silicon)
6. K: Kalium (Potassium)
7. Ca: Calcium (Calcium)
8. Ba: Barium (Barium)
9. Fe: Eisen (Iron)

1.2 Fragestellung

Diese neun Bestandteile der gefundenen Glassplitter sollen nach der Untersuchung in die oben genannten sieben Klassen eingeteilt werden. Somit soll es in Zukunft möglich sein die Glassplitter anhand der Eigenschaften und Bestandteile zu einem gewissen Glastypen zuweisen zu können.

Es handelt sich also um ein Klassifizierungsproblem.

1.3 Ziel der Analyse

Ziel der Analyse ist es, eine möglichst hohe Genauigkeit bei den Test-Durchläufen der Daten zu erhalten um eine wirkliche juristische Aussagekraft zu erzielen. Im Internet wurde eine vergleichbare Arbeit gefunden. Der Test-Score dieser Arbeit soll mindestens gleich hoch sein.

2. Rohdaten

Der zweite Abschnitt des Papers befasst sich mit den gewählten Daten. Dabei soll die Quelle und der Grund der Datenwahl erläutert werden. Außerdem wird auf die Größe und Form der Daten eingegangen.

2.1 Quelle

Die Daten kommen vom Forensic Science Service (FSS). Das FSS war eine britische Institution, welche die Polizeibehörden bei wissenschaftlichen Untersuchungen zu verschiedenen Straftaten unterstützte. Im Jahr 2012 wurde das FSS geschlossen und viele Datensätze wurden publik. (wikipedia.org, 2017)

Die Daten selbst kommen nicht direkt von der Homepage des FSS, sondern von einer Database. (Aha, 2017)

2.2 Datenwahl

Die Daten wurden gewählt, weil sie anschaulich sind und die später erläuterten Methoden sich so gut darstellen lassen. Außerdem sind es Daten welche in der Realität zum Einsatz kommen und das bei einem wichtigen Thema wie Verbrechensbekämpfung. Des Weiteren sind die Daten frei zugänglich und es können so Vergleiche zu anderen Projekten bezogen auf das Ergebnis gemacht werden.

2.3 Datengröße

Wie unter Punkt 1 bereits beschrieben, hat der Datensatz neun features und sieben targets. Insgesamt wurden 214 Glassplitter von verschiedenen Tatorten untersucht. Das heißt es gibt 10 Spalten und 214 Zeilen.

3. Verwendete Methoden

Im folgenden Abschnitt werden die verwendeten Modelle und Methoden erklärt und begründet warum sie benützt wurden. Es sollen aber auch Stärken und Schwächen der Modelle begründet werden. Am Beginn des Kapitels werden aber zuvor noch die Daten dargestellt.

3.1 Untersuchung der Daten

Der erste Schritt nach dem Einlesen des Datensatzes und laden der Bibliotheken ist es, die Daten zu untersuchen. Dabei wurde folgende Graphik erstellt.

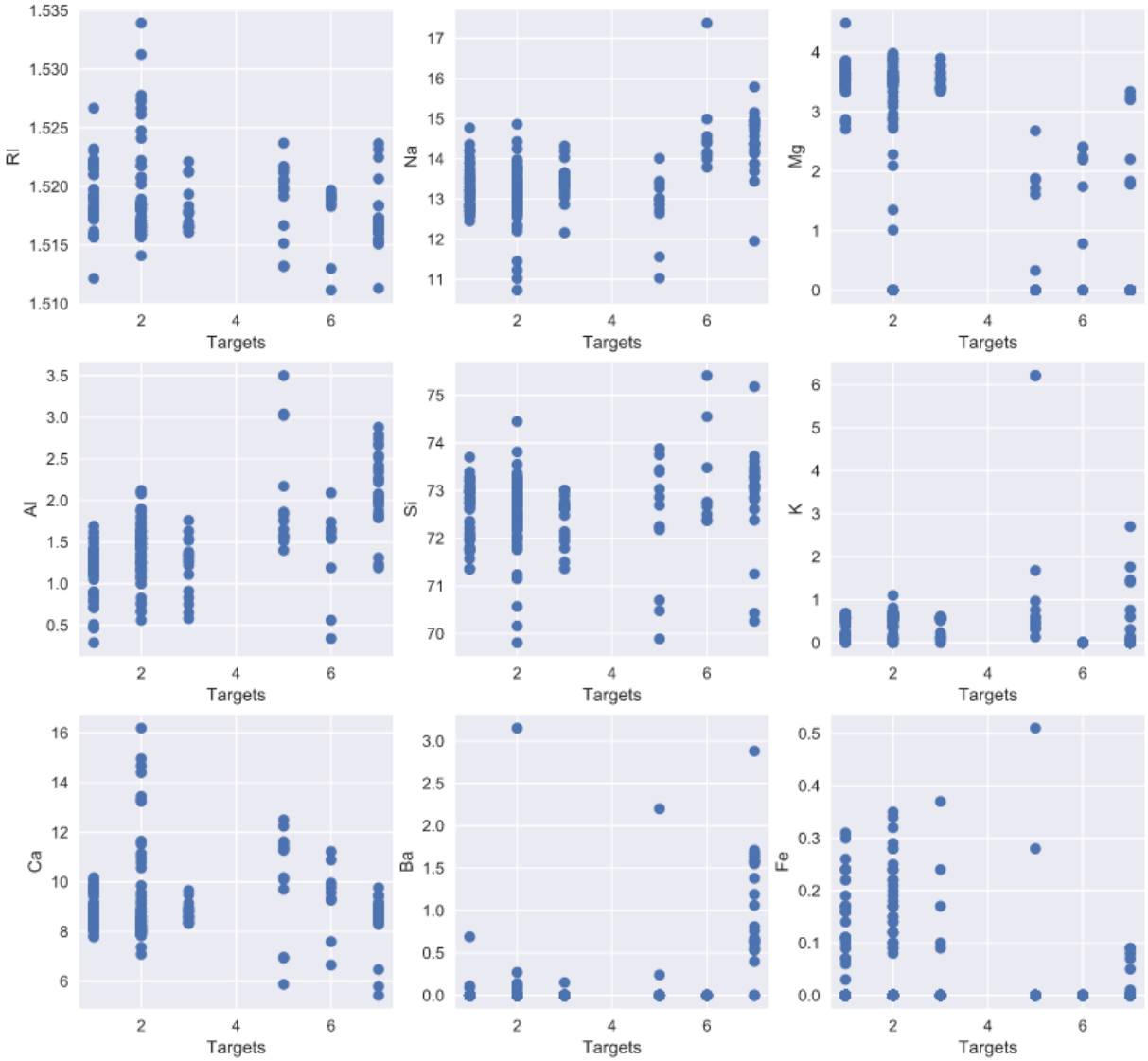

Abbildung 1: Plot der features über die targets.

Diese Graphik wird gewählt, um die Zusammenhänge der einzelnen features mit den jeweiligen targets darzustellen. Auf der X-Achse sind die sieben targets aufgereiht und auf der Y-Achse die jeweiligen features.

Es ist zu erkennen, dass die meisten features weit verstreut über die targets liegen. Einige wenige features sind bei einzelnen targets recht kompakt aufgelistet. Eine zusätzliche Korrelationsrechnung zeigt aber, dass keine wirklichen Zusammenhänge vorhanden sind.

3.2 k-Nearest Neighbors Classifier

Die erste untersuchte Methode ist der k-Nearest Neighbors Classifier. Er wird verwendet um ein Gefühl für das Projekt und die Daten zu bekommen. Da es hier nur die Anzahl der Nachbarn anzugeben gilt, zählt er zu den einfachsten Algorithmen der Klassifikation.

3.2.1 Vorgehensweise

Wie bei allen anderen Modellen auch, müssen die Daten in einen Trainings- und Testabschnitt aufgespalten werden. Danach wird der Algorithmus angewandt. Die Trainingsdaten helfen dabei das Modell zu erstellen und die Testdaten evaluieren eben dieses. Beim ersten Versuch wird eine beliebige Anzahl von Nachbarn gewählt. Da die Lösung dieser ersten Klassifikation nicht aussagekräftig ist, wird eine Schleife erstellt, in welcher die Anzahl der Nachbarn zwischen 1 und 10 variiert. Die Ausgabe wird um folgenden Plot erweitert.

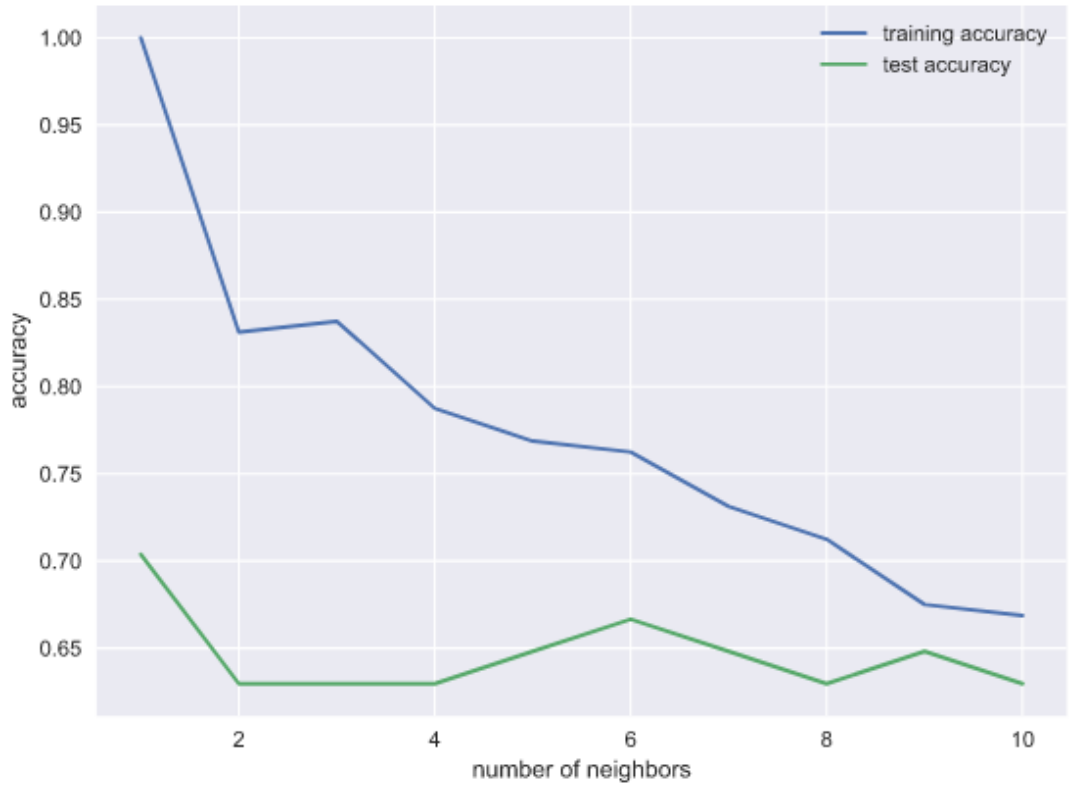

Abbildung 2: k-NN Classification, Ergebnisse der Test und Trainingsgenauigkeit bei 1-10 Nachbarn

Die blaue Linie zeigt die Trainingsgenauigkeit und die rote Linie die Testgenauigkeit. Man könnte overfitting vermuten, da die Genauigkeiten bei einem Nachbar weit auseinanderliegen, das liegt aber daran, dass der Algorithmus bei einem Nachbar im Trainingsset eben genau den richtigen auswählt, weil er ihn schon kennt und so eine

perfekte Trefferquote erzielt. Nichts desto trotz, erreicht man bei einem Nachbar eine gute Testgenauigkeit.

3.2.1.1 Grid Search mit Cross Validation

Grid Search ist eine Methode, welche die Parameter der Modelle so lange miteinander kombiniert, bis die Parameter mit den besten Ergebnissen gefunden sind. Das Cross Validation- Verfahren teilt die Daten in gleichmäßige Teile auf. Die Menge an Aufteilungen kann bestimmt werden. Dabei wird eine Teilmenge als Testdaten und alle anderen Teilmengen als Trainingsdaten genutzt. Danach wird die Teilmenge der Testdaten ausgetauscht und jede Teilmenge wird einmal als Testdatensatz verwendet. Dabei wird der Durchschnittswert berechnet und das Modell wird so verallgemeinert. Dadurch erhält man eine allgemeinere Aussage, da die Aufteilung der Daten nicht zufällig gut oder schlecht sein kann.

Die folgende Abbildung zeigt nun die durchschnittliche Testgenauigkeit mit Grid Search und Cross Vaidation.

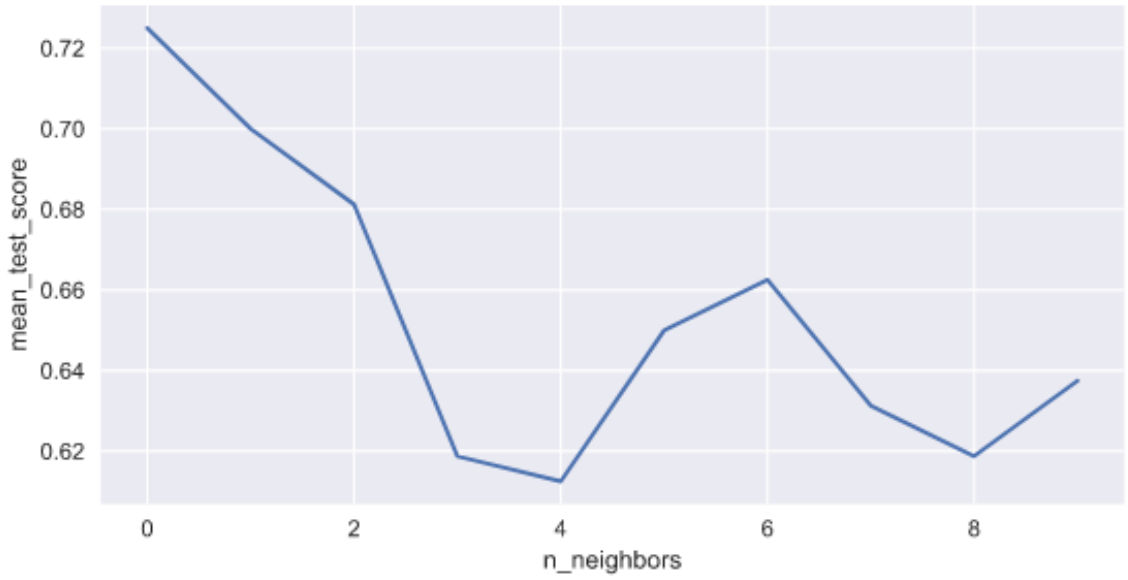

Abbildung 3: Durchschnittliche Testgenauigkeit mit Grid Search und Cross Validation

Auf der X-Achse sind die Nachbarn von 1 bis 10 aufgetragen und auf der Y-Achse die durchschnittliche Testgenauigkeit. Es ist zu erkennen, dass das beste Ergebnis bei einem Nachbarn liegt. Im Gegensatz zu Abbildung 2 wurde eine Steigerung des Scores erreicht.

3.2.1.2 Vorhersage mit der k Nearest Neighbors Prediction

Die kNN-Methode hat auch eine Methode zum Vorhersagen. Man kann also mit dem bereits Gelernten neue Daten klassifizieren. Da keine neuen Daten zu finden sind und eine Erstellung von neuen Daten wenig Sinn macht, werden die bereits vorhandenen Daten so aufgeteilt, dass der Algorithmus sie als neue Daten erkennt und eine Vorhersage macht. Dabei kommt man auf eine Vorhersagequote von 63 %. Das heißt,

wenn neue Glassplitter auf Tatorten gefunden werden, kann mit 63% Genauigkeit sagen, woher sie kommen.

3.2.1.3 Scaling

Da die Daten in sehr unterschiedlichen Skalen auftreten, werden sie nun skaliert um ein möglicherweise besseres Ergebnis zu bekommen. Beim Betrachten von Abbildung 1 ist zu erkennen, dass die Skalen von 0-0,5 beim feature Eisen gehen und von 70-75 beim feature Silizium. Was daher logisch ist, dass allgemeingebräuchliches Glas zum Großteil aus Silizium besteht. (wikipedia.org 2, 2017)

Es gibt verschiedene Scaler. Es wird der MinMax-Scaler, welcher die Skala der features auf Werte zwischen und 1 kürzt, und der Normalizer, welche die Arrays auf den Wert 1 bringt.

Die Ergebnisse der Testdaten verschlechtern sich bei beiden Skalierungen. Das war eigentlich schon zu erwarten, da der kNN nicht gut auf Skalierung anspricht, da sowieso nur die Abstände zu den Nachbarn verwendet werden.

3.2.1.4 PCA –Principal Component Analysis

Bei der PCA werden die Daten vereinfacht und strukturiert. Meist wird diese Methode bei sehr großen Datensätzen wie bei Datensätzen mit Bildern verwendet. Es sollen dabei Linearkombinationen gefunden werden und somit weniger Daten zum Rechnen verwendet werden. Dadurch beschleunigt sich die Analyse natürlich. Wie zu erwarten, ist keine Verbesserung der Ergebnisse eingetroffen, da die Daten nicht gut korrelieren. Allerdings kann man mit nur 2 x 9 Komponenten einen Score von 57 % erreichen.

3.3 Logistische Regression

Prinzipiell gilt bei der logistischen Regression eine analoge Vorgehensweise wie bei den anderen Modellen.

Die logistische Regression ist ein lineares Modell. Besonders zu beachten gilt der Parameter C. Dieser reguliert den Algorithmus umso mehr, je kleiner das C gewählt wird. Deshalb wurden verschiedene Größen für C ausprobiert. Dabei zeigt sich, dass bei einem groß gewählten C also bei geringer Regulierung ein Test-Score von knapp 58 % erreicht werden kann. Bei einem sehr klein gewählten C fällt der Test-Score auf knapp 45%. Folgende Abbildung gibt diese wieder. Es wurde eine Große Spannweite bei der Größe von C gewählt um die unterschiedliche Ergebnisse darzustellen.

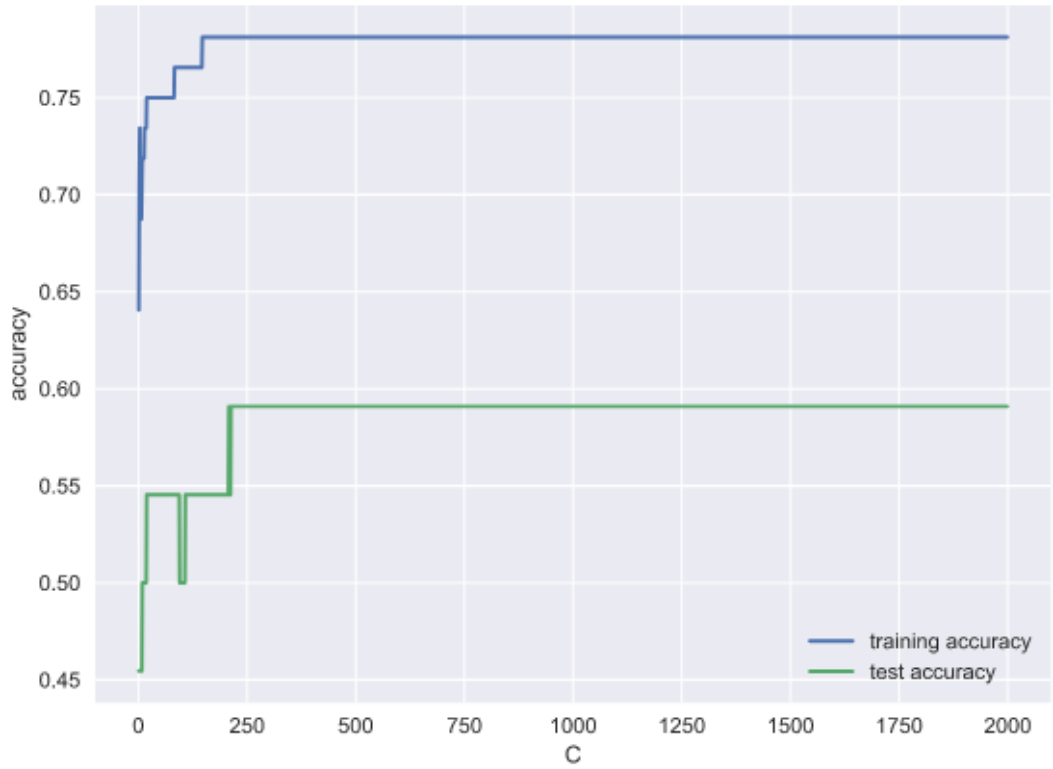

Abbildung 4: Scores der logistischen Regression in Abhängigkeit von Parameter C

Wie auch bei den vorherigen Modellen wird ein Grid -Search mit CV gemacht. Also werden wieder die besten Parameter gesucht und der Zufall eliminiert. Folgende Graphik veranschaulicht die Ergebnisse.

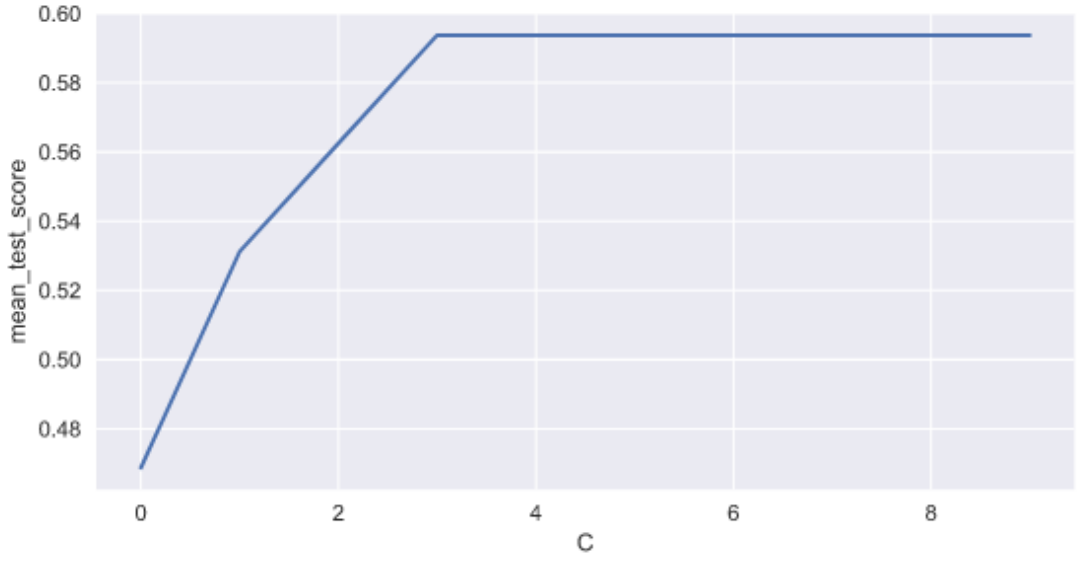

Abbildung 5: Durchschnittlicher Score bei der logistischen Regression

Es ist zu erkennen, dass ein durchschnittlicher Test-Score von knapp 60% erreicht werden kann. Eine Steigung zum händischen ausprobieren der Parameter also.

3.3.1 Stochastic Gradient Descent Classifier (SGDC)

Der SGDC ist ein Verfahren, welches durch Iteration ein Maximum beziehungsweise Minimum sucht. Hauptsächlich wird dieses Verfahren bei sehr großen Datenmengen verwendet. Der Vorteil ist, dass es viele Parameter, also viele Stellschrauben zum Drehen gibt. Wie bei den anderen Modellen wird wieder ein einzelner Testversuch mit den voreingestellten Parametern gemacht. Hierbei wird ein kleiner nicht zufriedenstellender Score erreicht. Deshalb wird wieder ein Parametertuning in Form von Grid Search und eine Cross Validation gemacht. Der durchschnittliche Score dabei ist etwas höher, aber noch immer nicht besonders gut. Die folgende Abbildung zeigt dies.

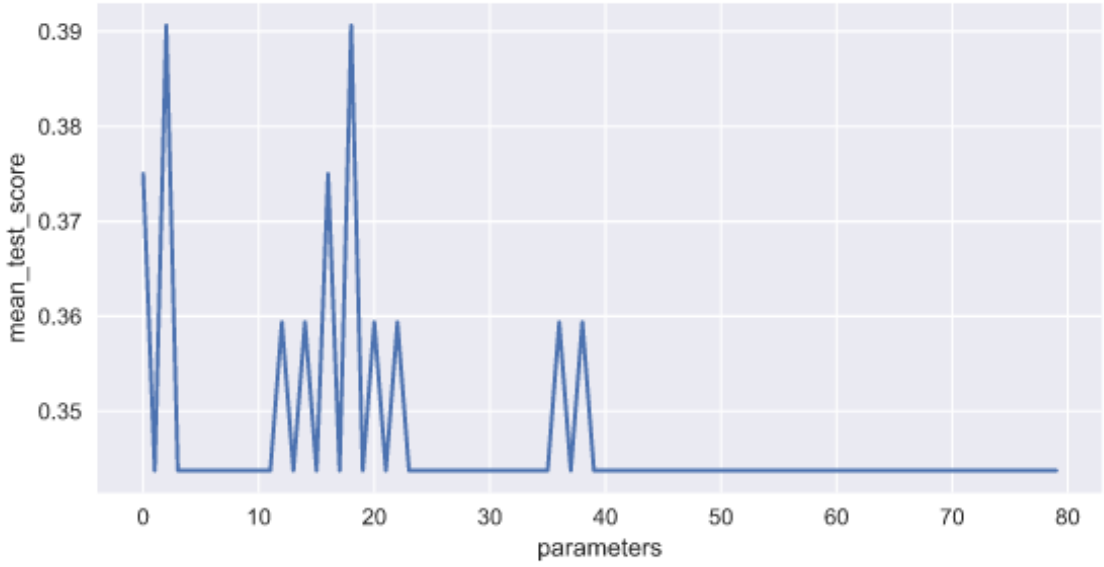

Abbildung 6: Durchschnittlicher Score beim SGD Classifier

3.3.1.1 Scaling

Wie auch beim k-NN wurde die Skalierung durchgeführt. Die Scores sind dabei eindeutig angestiegen, da der SGD auf Grund von internen Abläufen sensibel darauf anspricht. Gesamtheitlich gesehen ist der Score zwar immer noch zu gering. Aber es ist eine Verbesserung zu erkennen.

3.3.1.2 Kernel Approximation

Die Kernel-Approximation wurde genutzt, da sie laut auf die Daten zugeschnitten ist. (scikit-learn developers, 2017)

Bei der Kernel Approximation werden nichtlineare Eingangsparameter so transformiert, dass sie als Grundlage von linearen Klassifizierungs-Methoden dienen.

Bei der Kernel Approximation werden jene Daten ausgewählt welche das Beste Nutzen zu Aufwand Verhältnis haben. Die Methode ist also grundsätzlich ähnlich zur PCA. Durch diese Methode kann ein Score von über 80 % erreicht werden. Der eindeutig beste Score der bisher erreicht wurde.

3.3.1.3 Monte Carlo Simulation

Bei mehrmaliger Ausführung der Code-Zeile fällt auf, dass die Ergebnisse stark variieren. Deshalb wurde eine Monte-Carlo Simulation gemacht. Diese dient dazu, das Ergebnis zu verifizieren. Bei fünfhundert Versuchen wurde der Mittelwert berechnet. Dieser ist mit 85 % sehr gut. Das untenstehende Bild zeigt die Monte Carlo Simulation.

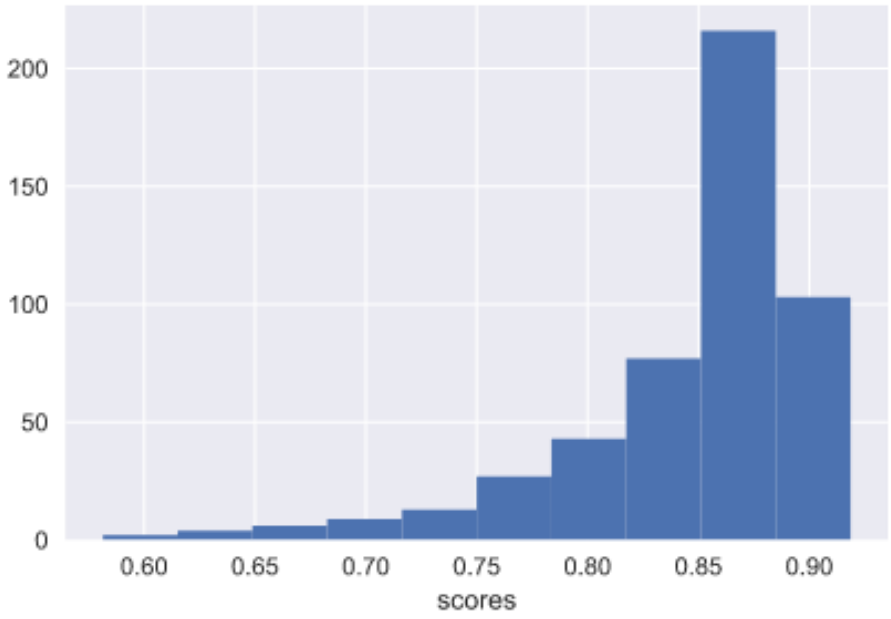

Abbildung 7: Monte Carlo Simulation nach der Kernel Approximation

4. Reflexion

Im folgenden Abschnitt der Projektarbeit wird nochmals auf die behandelten Methoden und Modelle eingegangen. Dabei sollen die Ergebnisse analysiert werden und ein Vergleich mit einem anderen Projekt gemacht werden.

4.1 Reflexion von k-NN

Die Verwendung von k-NN ist deshalb nachvollziehbar, da die Einfachheit des Modells Einblick in Data Science gibt. Die Ergebnisse nach dem ersten Testversuch waren vielversprechend, dass die Daten gut gewählt sind. Nach dem Grid-Search und der Cross Validation haben sich die Ergebnisse verbessert.

Da die Skalen der unterschiedlichen features stark variieren war das Scaling der nächste logische Schritt. Die Ergebnisse wurden bei allen Scalern schlechter, dies liegt aber wie oben beschrieben am k-NN Algorithmus. Auch die PCA lieferte insgesamt schlechtere Ergebnisse. Insgesamt deshalb, da durch die PCA mit einer geringeren Anzahl von Daten ein recht guter Score erzielt werden konnte. Da, wie erwähnt, die PCA normalerweise bei Datensätzen mit sehr viel mehr Daten zur Anwendung kommt, ist das erlernte für mögliche spätere Arbeiten sehr vielversprechend.

4.2 Reflexion der logistischen Regression

Die logistische Regression wurde deshalb gewählt, weil es der Parameter C zulässt, die wichtigsten Datenpunkte mehr zu beachten. Dies geschieht bei der Wahl von kleinen C-Größen. Allerdings wurde bei einem neutralen C-Wert die besten Ergebnisse erzielt. Der Datensatz braucht also Regulation, allerdings nicht zu viel und auch nicht zu wenig.

4.3 Reflexion der SGDC

Da die Ergebnisse bis dato nicht wirklich zufriedenstellend waren, wurde eine Internetrecherche gestartet. Auf scikit-learn.org wurde dann die oben bereits erwähnte Karte für Data Science gefunden. Die Ergebnisse des vielversprechende SGD-Classifier, waren aber auch nach dem Parameter-Tuning schlechter als die bereits erreichten.

Da das Scaling beim k-NN nicht funktioniert hat, wurde es auch bei diesem Modell nochmals versucht. Mit dem Scaler wurden bessere Scores erreicht. Dann wurde die Kernel-Approximation gemacht. Diese lieferte die besten Ergebnisse.

4.4 Vergleich mit der Literatur

Natürlich sagen die Ergebnisse allein nicht viel aus. Um zu wissen ob man gut gearbeitet hat, benötigt man Vergleiche. Da die Daten schon länger im Netz kursieren, wurde auch

ein entsprechendes Notebook gefunden. Der beste Score von diesem liegt bei 77,5. (Kawerk, 2017)

Literaturverzeichnis

Aha, D. (25. 06 2017). *archive.ics.uci.edu*. Von https://archive.ics.uci.edu/ml/datasets/glass+identification abgerufen

Kawerk, E. (28. 06 2017). *kaggle.com*. Von https://www.kaggle.com/eliekawerk/glass-type-classification-with-machine-learning abgerufen

scikit-learn developers. (27. 06 2017). *scikit-learn.org*. Von http://scikit-learn.org/stable/tutorial/machine_learning_map/index.html abgerufen

wikipedia.org 2. (27. 06 2017). *wikipedia.org*. Von https://de.wikipedia.org/wiki/Glas abgerufen

wikipedia.org. (06. 27 2017). *wikipedia.org*. Von https://en.wikipedia.org/wiki/Forensic_Science_Service abgerufen