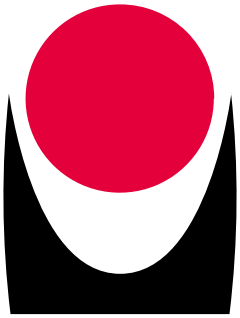

TECHNISCHE HOCHSCHULE LÜBECK

Technische Hochschule Lübeck
Fachbereich Informatik und Softwaretechnik
Informatik / Softwaretechnik, B.Sc.

Intelligente Systeme

Autor	**Matr.-Nr.**
Tim Lüneburg	321226
Denis Alipkina	326771

Wintersemester 2021 & 2022
Lübeck - 19. Juni 2023

Inhaltsverzeichnis

Abbildungsverzeichnis

1. Auffinden von Strukturen auf Basis unsicherer Information

1.1. Ausgangssituation und Zielsetzung

Bei dieser Aufgabe geht es darum, eine Tätigkeit zu automatisieren, die sehr zeitaufwändig ist, wenn sie von Hand durchgeführt wird: In großen Datensätzen sollen bestimmte Strukturen identifiziert und markiert werden, von denen wir lediglich wissen, dass sie lokale Maxima darstellen. Das Problem dabei ist, dass keine genaue Definition der gesuchten Strukturen vorliegt. Für einen menschlichen Betrachter, der mit der Bedeutung der Daten vertraut ist, ist es dennoch prinzipiell sehr einfach, die gewünschten Strukturen zu identifizieren. Als Basis für die Entwicklung einer automatisierten Lösung stehen hier deshalb neben zwei Datensätzen (data0.csv und data1.csv), die als zweidimensionale Matrizen mit Höhenwerten dargestellt sind, jeweils eine Liste mit den x/y-Koordinaten der von Hand markierten lokalen Maxima (label0.csv und label1.csv) zur Verfügung. Die x/y-Koordinaten geben die Zeile bzw. die Spalte (beginnend mit 0) in der Matrix an. Alle Daten liegen im CSV-Format vor. Trennzeichen ist ein Komma.

1.2. Aufgabenstellung

1. Untersuchen Sie den Datensatz 0 (also data0.csv) und versuchen Sie, Kriterien zu ermitteln, mit denen sich die zugehörigen Markierungen (label0.csv) erklären lassen.

2. Entwickeln Sie einen Algorithmus, der die Markierung der Datensätze, also das Auffinden der gesuchten Strukturen, automatisiert.

3. Evaluieren Sie Ihren Algorithmus anhand der manuellen Label (label0.csv) und bestimmen Sie Precision, Recall und F-Score für Ihren Algorithmus.

4. Verbessern Sie die Leistung ihres Algorithmus. Verwenden Sie für Ihre Optimierung nur den Datensatzes 0, um eine möglichst unabhängige Evaluation durchführen zu können. Im Endergebnis sollte auf dem Datensatz 1 (also data1.csv) ein F-Score von mindestens 0,8 angestrebt werden.

1.3. Hinweise

Der Recall ist der Quotient von der Anzahl der vom Algorithmus korrekt gefundenen Label und der Anzahl der tatsächlich vorhandenen Label.

Die Precision ist der Quotient von der Anzahl der vom Algorithmus korrekt gefundenen Label und der Gesamtanzahl der vom Algorithmus gefundenen Label.

Precision und Recall besitzen den Wertebereich 0...1.

Der F-Score ist der harmonische Mittelwert von Precision und Recall.

Ein Label gilt als "korrekt gefunden", wenn seine Koordinaten mit denen eines Labels aus der Datei label0.csv (bzw. label1.csv) übereinstimmen oder einen Punkt in dessen Nachbarschaft markieren, der dieselbe Höhe besitzt. Bei einer Evaluation darf jedes Label aus der Datei label0.csv (bzw. label1.csv) jedoch höchstens einmal gezählt werden, d.h. weitere unmittelbare Nachbarpunkte dürfen nicht als weitere korrekt gefundene Labels gezählt werden.

2. Aufgabe 1

Als erstes haben wir die *Data0.csv* über Python eingelesen und haben die in Abbildung 2.1 Heatmap der Data0.csv zu sehene Heatmap erhalten. Hier ist zu erkennen dass es sich bei den lokalen Maximas um eine Art Kreisform handelt. Die Vermutung liegt nahe, dass der Radius um das jeweilige lokale Maximum eine Rollen spielen könnte. Ein weiterer Gedanke war, dass die Steigung an lokalen Maximas der *Label0.csv* zu den anderen unterscheidet. Ebenfalls kam die Idee auf, dass man die Höhe der lokalen Maximas betrachten beziehungsweise diese im Zusammenhang mit dem Radius anschaut. Die Überprüfung der Umgebung der lokalen Maximas auf Symetrie, wäre ebenfalls eine Überlegung wert, da wir hier keinen Parameter benötigen.

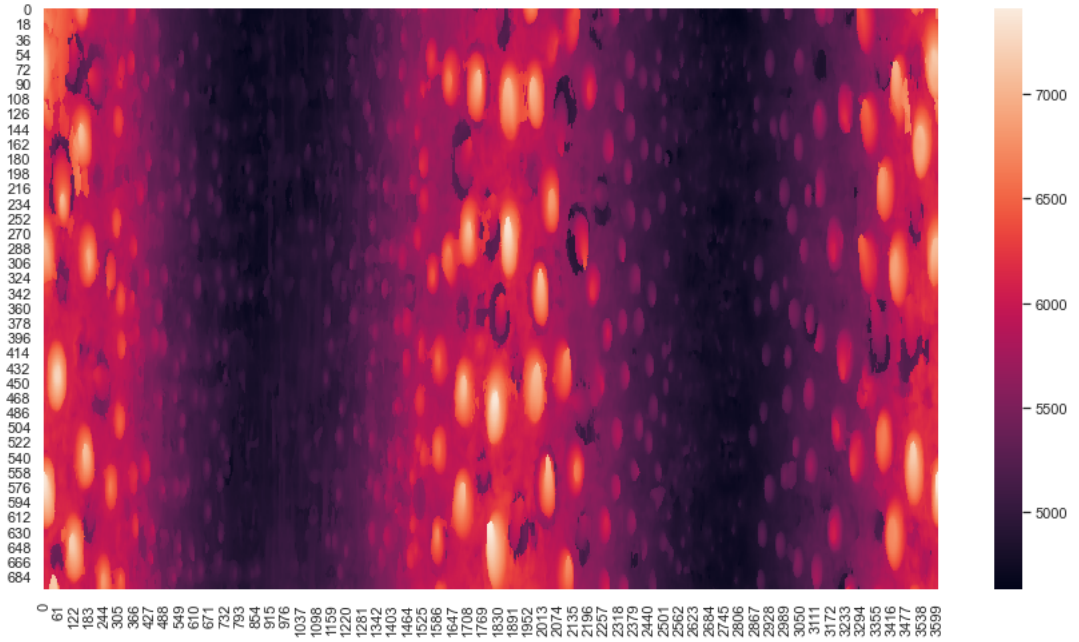

Abbildung 2.1.: Heatmap der Data0.csv

3. Aufgabe 2

Im nächsten Schritt haben wir die *Data0.csv* und die *Label.csv* eingelesen und haben verteilt auf der Vertikalen (Rows) Agenten gespawnt und haben diese alle lokalen Maximas der *Data0.csv* suchen lassen. In der Abbildung 3.1 Spawnpunkte der Agenten ist zu sehen dass wir 4 Agenten spawnen lassen haben und diese gleichverteilt gespawnt werden und Start und Ende des Agenten jeweils markiert sind. Die Zahlen 9 stellen hier einfach nur ausgedachte Daten der *.csv* Datei dar um die Abbildung zu vereinfachen.

		0	1	2	3	..	2699
Agent 1 Start	0	9	9	9	9	..	9
	.	9	9	9	9	..	9
	.	9	9	9	9	..	9
Agent 1 Ende	174	9	9	9	9	..	9
Agent 2 Start	175	9	9	9	9	..	9
	.	9	9	9	9	..	9
	.	9	9	9	9	..	9
Agent 2 Ende	349	9	9	9	9	..	9
Agent 3 Start	350	9	9	9	9	..	9
	.	9	9	9	9	..	9
	.	9	9	9	9	..	9
Agent 3 Ende	524	9	9	9	9	..	9
Agent 4 Start	525	9	9	9	9	..	9
	.	9	9	9	9	..	9
	.	9	9	9	9	..	9
	.	9	9	9	9	..	9
Agent 4 Ende	699	9	9	9	9	..	9

Abbildung 3.1.: Spawnpunkte der Agenten

Dabei haben wir insgesamt **33033** Lokale Maximas gefunden. Dabei ist uns aufgefallen, dass bei den **443** Labeln der *Label0.csv* diese **3** folgenden Label **keine** Lokalen Maximas sind:

1. Row: 137 Column: 307
2. Row: 267 Column: 617
3. Row: 350 Column: 2550

In dem Hinweise der Aufgabe war beschrieben, wann ein gefundenes Label als korrekt gilt. Deshalb haben wir dann die lokalen Maximas als Plateau zusammengefasst und haben die Anzahl von lokalen Maximas auf **3754** reduziert. Dann ist uns ebenfalls aufgefallen, wenn wir die lokalen Maximas als Plateau zusammengefassen, gibt es folgendes weitere Label: Row: 224 Column: 1419 aus der *Label0.csv*, welches kein lokales Plateau Maximum ist. Wir haben damit **4** Label gefunden die eigentlich keine Label abbilden sollten und haben diese anschließend aus der *Label0.csv* entfernt.

Für die weitere Implementierung haben wir uns entschieden auf den Radius und die Steigung zu fokussieren. Dazu haben wir Informationen über den Radius und den Durchschnittswert für den

minimalen Wert aus dem Umkreis der Label sowie nicht Label herausgefunden. Um die Informationen zu verarbeiten, nutzen wir DescriptiveStatistics und erhalten folgende Werte.

Label:	**Radius**	**Avg. Minimal Wert**
n:	439	10579
min:	1.0	0.0
max:	207.0	2342.0
mean:	24.278	511.765
std dev:	28.7669	463.929
median:	19.0	381.0
skewness:	1.932	1.6962
kurtosis:	6.744	2.5265

Nicht Label:	**Radius**	**Avg. Minimal Wert**
n:	3315	11327
min:	1.0	0.0
max:	130.0	1788.0
mean:	3.467	122.237
std dev:	5.809	247.045
median:	1.0	32.0
skewness:	7.545	3.821
kurtosis:	105.651	17.301

Die hier markierten Mittelwerte sind für unseren Algorithmus interessant und betrachten wir als Start Schwellwerte für die Evaluierung.

4. Aufgabe 3

Wenn die Agenten die lokalen Plateau Maximas gefunden haben, dann erhalten wir insgesamt **3754** lokale Plateau Maximas, wie in Abbildung 4.1 Korrekte und Inkorrekte Label im Vergleich zu sehen ist sind davon **439 Korrekte Label** und **3315 Nicht Korrekte Label**. Damit erhalten wir initial vor der Anwendung des Algorithmuses zum Filtern korrekter Label mit Radius und Steigung folgende Werte:

Precision:	0,1169
Recall:	1,0
F-Score:	0,2093

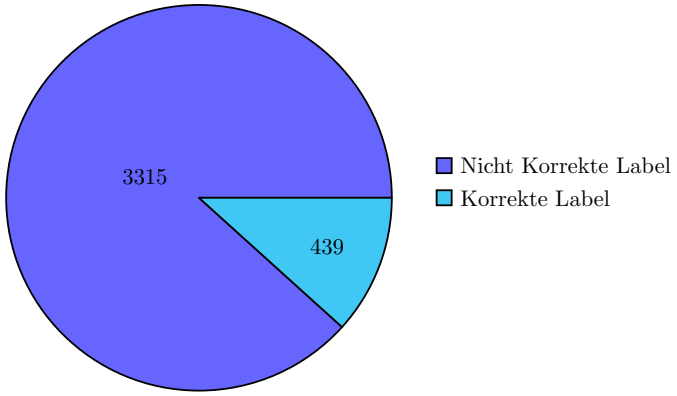

Abbildung 4.1.: Korrekte und Inkorrekte Label im Vergleich

Nach Ausprobieren mit verschiedenen Schwellwerten für den Radius haben wir mit dem **Schwellwert: 18** das beste Ergebnis von insgesamt **584** Label erhalten. Davon sind wie in Abbildung 4.2 Korrekte und Inkorrekte Label im Vergleich nach dem Filter Radius zu sehen **187** falsch und **397** korrekt. Damit erhalten wir die neuen folgenden Werte:

Precision:	0.679795
Recall:	0.904328
F-Score:	0.776149

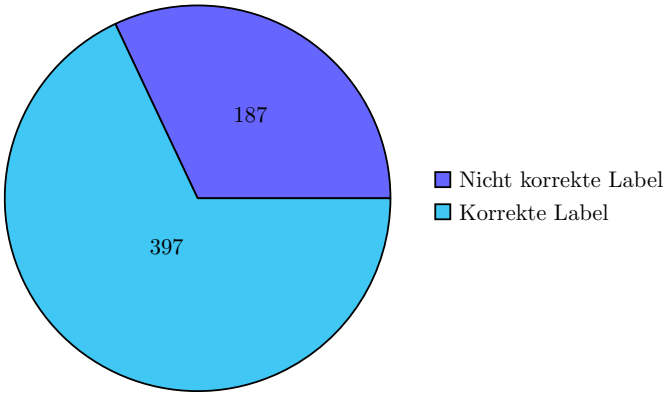

Abbildung 4.2.: Korrekte und Inkorrekte Label im Vergleich nach dem Filter Radius

Um den Algorithmus noch etwas zu verbessern, haben wir noch einen weiteren Filter für die Steigung hinzugefügt. Dort haben wir ebenfalls verschiedene Schwellwerte ausprobiert und haben mit dem **Schwellwert: 134** das beste Ergebnis von insgesamt **480** Label erhalten. Davon sind wie in Abbildung 4.2 Korrekte und Inkorrekte Label im Vergleich nach dem Filter Radius zu sehen **102** falsch und **178** korrekt. Damit erhalten wir die neuen folgenden Werte:

Precision:	0,787500
Recall:	0,8610488
F-Score:	0,822633

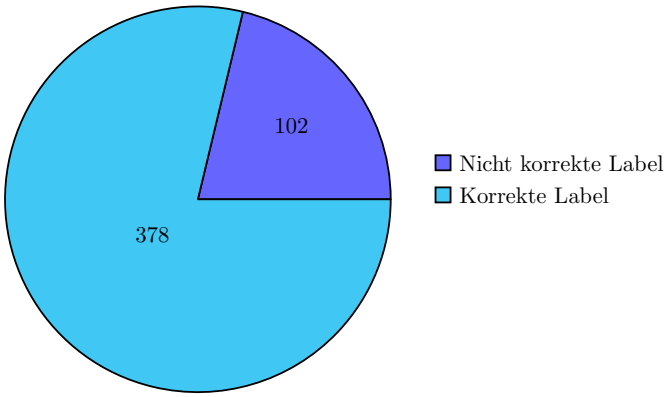

Abbildung 4.3.: Korrekte und Inkorrekte Label im Vergleich nach dem Filter Radius

5. Aufgabe 4

Um das bestehende Verfahren zu verbessern, haben wir uns vorgenommen eine Klassifizierung zu implementieren. Statt erst die gefundenen Maximas zu filtern und anschließend die gefilterten Daten erneut zu filtern, sollen beide Filter gleichzeitig angewendet werden. Dies schließt aus, dass Labels die in der Nähe einen Nachbarn haben nicht direkt rausgefiltert werden, sondern noch eine zweite Chance bekommen und deren Steigung überprüft wird.
In unserer Implementierung läuft der Agent über die Landschaft und wenn er ein lokales Maximum findet, wertet er die gefunden Stelle mit beiden Kriterien aus. Die Auswertung ist jeweils eine Zahl zwischen 0 und 1 was wie **nicht sicher** und **sehr sicher** zu verstehen ist. Werden nun beide Auswertungen multipliziert, kriegt man auch eine Zahl zwischen 0 und 1. Anhand der errechneten Zahl, beurteilt dann der Agent ob es sich um ein potenzielles Label handelt oder nicht.
Natürlich gibt es beim Agenten einstellbare Parameter die als Schwellwerte für die Auswertung genutzt werden. In dieser Doku erklären wir nun einen der Filter beziehungsweise Auswertungsmethoden.
Um die Steigung auswerten zu bekommen, benutzt der Agent zwei Parameter. Der erste Parameter gibt an ab welcher berechneten Steigung er sich zu 50 Prozent sicher sein soll und der zweite Parameter gibt den Schwellwert für 75 prozentige Sicherheit an. Anhand dieser Schwellwerte wird eine liniare Funktion f(x) erstellt. Um eine errechnete Steigung auszuwerten, wird diese als x in die Funktion geben und man bekommt eine Sicherheit zwischen 0 und 1.
Am Anfang haben wir per Hand die Werte eingestellt und kamen mit den Parametern **Schwellwert für 50 prozentige Sicherheit = 80** und **Schwellwert für 75 prozentige Sicherheit = 102** auf optimale Ergebnisse. Damit das Auffinden von Strukturen unabhängig und automatisiert funktioniert, mussten wir herausfinden warum genau diese Parameter optimal sind. Wir haben dann alle Filter rausgenommen, sodass alle lokalen Maximas als potenzielle Labels erkannt wurden. Diese haben wir dann in korrekte und nicht korrekte aufgeteilt und die Methode zur Berechnung der Steigung angewandt. Wir bekammen folgende Ausgaben:

Korekkte Labels:
Radius: 1, Min: 0, Avg: 2,489749, Max: 16
Radius: 5, Min: 5, Avg: 40,487472, Max: 324
Radius: 10, Min: 20, Avg: 105,503417, Max: 560
Radius: 15, Min: 34, Avg: 161,947608, Max: 653
Radius: 20, Min: 40, Avg: 204,840547, Max: 775
Radius: 25, Min: 38, Avg: 235,908884, Max: 828
Radius: 30, Min: 34, Avg: 258,551253, Max: 983

Inkorekkte Labels:
Radius: 1, Min: 0, Avg: 1,862142, Max: 99
Radius: 5, Min: -246, Avg: 19,250980, Max: 347
Radius: 10, Min: -362, Avg: 29,326697, Max: 508
Radius: 15, Min: -413, Avg: 28,739668, Max: 457
Radius: 20, Min: -448, Avg: 24,407541, Max: 589
Radius: 25, Min: -473, Avg: 18,878431, Max: 713
Radius: 30, Min: -500, Avg: 12,853997, Max: 824

Wir hatten für die Berechnung des Radiuses 20 benutzt, da dieser die besten Werte lieferte. Beim betrachten der genannten Ausgabe fällt auf, dass sich die Mindest- und Durchschnittswerte der korrekten Label stark von den inkorrekten unterscheiden. Außerdem fällt auf, dass bei den korrekten Labels bei dem Radius 20 das Minimum am höchsten ist. Unser **Schwellwert für 50 prozentige Sicherheit** ist doppelt so hoch wie der Minimumwert beim Radius 20 und der **Schwellwert für 75 prozentige Sicherheit** die Hälfte vom Durchschnittswert beim Radius 20 ist. Aufgrund der knappen Zeit kamen wir nicht dazu die Automatisierung zu implementieren. Jedoch ist die genannte Rechnung hier angegeben und könnte so auch programmiert werden. Hierbei ist zu beachten, dass das gezeigt Verfahren in diesem Beispielt gut klappt, aber auch Zufall sein könnte, sodass diese vorerst mit Vorsicht zu beachten ist.
In unserem Fall haben wir fixe Werte, welche zu einer **Verbesserung von ca. 0,023 im F-Score** führen. Dadurch finden wir auf *data0.csv* insgesamt **448** Label, davon sind wie in Abbildung 5.1 Korrekte und Inkorrekte Label im Vergleich nach dem neuen Filter Verfahren auf Data0 zu erkennen **73** keine korrekten Label und **375** korrekte Label enthalten. Damit ergeben sich bei der Evaluierung folgende Werte:

Precision:	0,837054
Recall:	0,854214
F-Score:	0,845547

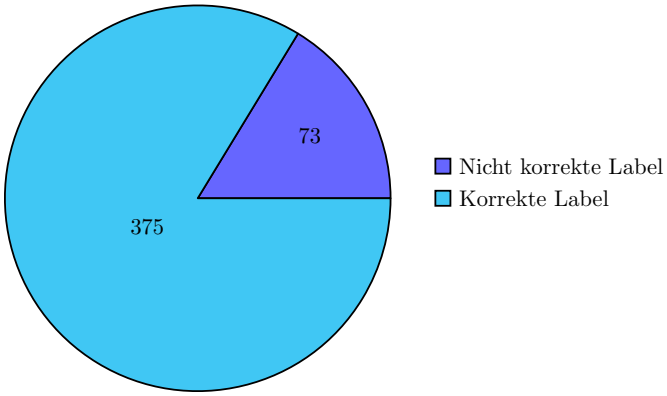

Abbildung 5.1.: Korrekte und Inkorrekte Label im Vergleich nach dem neuen Filter Verfahren auf Data0

Man erkennt, dass sich fast nur die Precision verbessert hat.
Welche weiteren Vorteile bietet die Klassifizierung?
Man könnte noch weitere Filter wie Symetrie einbauen. Die Anzahl der Filter könnte nach belieben beliebig groß sein. Jedoch muss man im Hinterkopf behalten, dass man damit **overfitten** könnte.

Wenn wir unseren Algorithmus mit den selben Parameter auf der *Data1.csv* anwenden, finden unsere Agenten insgesamt **233** Label, davon sind wie in Abbildung 5.2 Korrekte und Inkorrekte Label im Vergleich nach dem neuen Filter Verfahren auf Data1 zu erkennen **200** korrekte Label und **33** nicht korrekte Label gefunden worden. Damit erhalten wir bei der Evaluierung folgende Werte:

Precision:	0,858369
Recall:	0,865801
F-Score:	0,862069

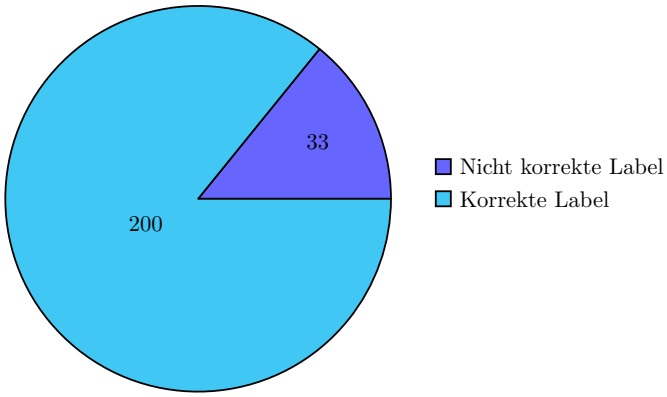

Abbildung 5.2.: Korrekte und Inkorrekte Label im Vergleich nach dem neuen Filter Verfahren auf Data1

A. Anhang

A.1. Aufbau des Codes

A.1.1. Landschaft

Die Klasse *Landschaft.java* umfasst die eingelesene Liste von DatenLeserStelle und speichert gefundene Label in unterschiedlichen Listen zwischen. Ebenfalls lässt die Landschaft eine in der Landschaft gesetzte Anzahl von Agent spawnen.

A.1.2. Agent

Die Klasse *Agent.java* ist ein Agent der sich auf der Landschaft befindet. Die Hauptaufgabe des Agenten ist es in einem bestimmten Bereich in der Landschaft zu laufen und Lokale Maximas bzw. Lokale Plateau Maximas zu finden.

A.1.3. Stelle

Die Klasse *Stelle.java* umfasst eine Stelle aus der Landschaft und beinhaltet Informationen über die einzelne Stelle und Aktionen die für eine Stelle ausgeführt werden können.

A.1.4. DatenLeserStelle

Die Klasse *DatenLeserStelle.java* liest Daten aus einer *Data.csv* ein und speichert die dort enthaltenen Werte in einer Liste von Listen mit Stelle.

A.1.5. LabelLeser

Die Klasse *Labelleser.java* liest eine *Label.csv* Datei ein und speichert diese in einer Liste von Stelle.

A.1.6. Schwellwerte

Die Klasse *Schwellwerte.java* ist eine Record-Klasse. Diese enthält nur die Schwellwerte die für Berechnungen benötigt werden.

A.1.7. Labelprüfer

Die Klasse *Labelpruefer.java* dient der Auswertung der Werte aus der Landschaft. Dabei werden hier die Label aus einer *Label.csv* hinzugezogen und verschiedenste Parameter berechnet.

A.1.8. Evaluation

Die Klasse *Evaluation.java* beinhaltet die Formel für die Berechnung von Precision, Recall und F-Score in drei methoden, welche die Berechnung anhand der Formel durchführen.

A.2. Falsche Label in der Label0.csv

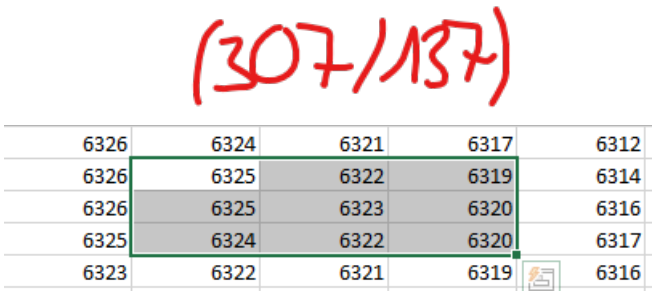

Abbildung A.1.: Row: 137 Column: 307

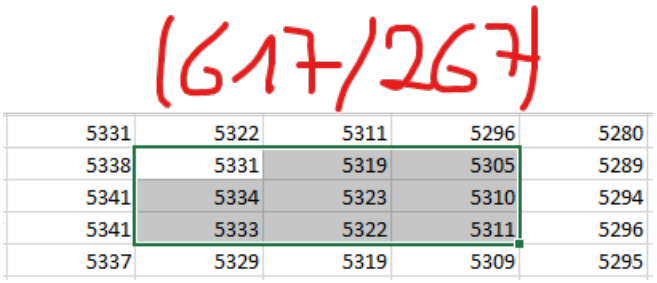

Abbildung A.2.: Row: 267 Column: 617

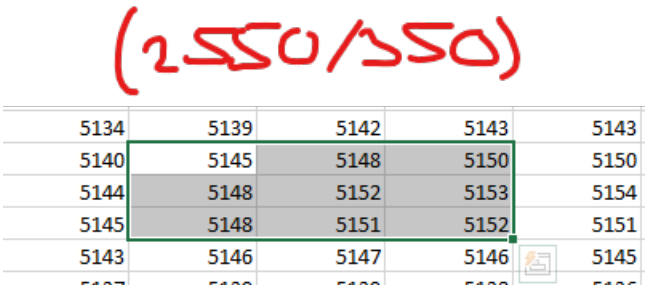

Abbildung A.3.: Row: 350 Column: 2550

(1419 | 224)

5555	5554	5553	5551	5549	5546	5543
5559	5559	5558	5556	5554	5550	5547
5560	5561	5560	5560	5557	5553	5550
5560	5561	5561	5561	5558	5555	5552
5556	5559	5561	5561	5559	5557	5554
5551	5557	5560	5561	5561	5559	5556
5545	5557	5561	5562	5563	5561	5557
5311	5558	5562	5563	5564	5562	5558
5314	5557	5562	5564	5564	5562	5559
5315	5551	5560	5564	5565	5563	5562
5317	5547	5560	5564	5565	5564	5563
5319	5552	5561	5564	5565	5565	5563
5323	5555	5561	5564	5564	5564	5562

Abbildung A.4.: Row: 224 Column: 1419