-

Auswertung von Eye Tracking Daten mittels Überführung von Metriken in Java-Objekte

Autor: Maximilian Jaschkowitz

1 Einleitung

Eye-Tracking ist eine interessante Methode in wissenschaftlichen und wirtschaftlichen Bereichen. Hierbei werden Bewegungen des Auges bei der Durchführung unterschiedlicher Tätigkeiten gemessen und Informationen darüber gesammelt, was und in welcher Form das Auge beobachtet.

Pramodini et al. (2017) beschreibt das Eye-Tracking als Messung der Blickpunkte, die unser Auge relativ zur Position des Kopfs erzeugt. Hierbei wird der Ort, der betrachtet wird, sowie der Pfad der Bewegung des Blicks gemessen und aufgezeichnet. Diese Beobachtungen können dann analysiert, visualisiert und interpretiert werden.

Um die Beobachtungen zu analysieren ist es notwendig, den gelesenen Text in sogenannte „Areas of Interest“ (AOI) zu unterteilen. Dies sind von den Autoren vordefinierte Felder im Text, welche einen bestimmten Inhaltsgehalt besitzen und daher vom Autor erfasst und verstanden werden müssten.

Während die Grundlage des Eye-Tracking Konzepts meist unverändert ist, gibt es Unterschiede in der Interpretation und Auswertung der Messungen. Ein relevantes Feld in dem Eye-Tracking zur Anwendung kommt, ist auch das Software Engineering.

Sharafi et al. (2015) haben die Anwendung des Eye-Tracking Verfahrens im Rahmen von Software Engineering aus unterschiedlichen Studien in ihrer Studie untersucht und die resultierenden Tracking Metriken verglichen und gegenübergestellt. Hierbei haben sie auch Definitionen für allgemeine Metriken herausgearbeitet.

Für diese Ausarbeitung werden aus jenen allgemeinen Metriken des Eye-Trackings einige Metriken ausgewählt und zur Durchführung einer deskriptiven Statistik und einer Regressionsanalyse verwendet. Dazu werden für die ausgewählten Metriken in Java Objekten erstellt.
Diese Objekte können auf Daten mehrerer Eye-Tracking-Messung zugreifen, welche ebenfalls in Form von Objekten in Java zur Verfügung gestellt werden.
Anschließend werden entsprechende Metriken für alle Probanden berechnet, deskriptiv und mittels einer Regressionsanalyse statistisch untersucht.

Die zu untersuchenden Eye-Tracking Daten stammen von zwei unterschiedlichen Probandengruppen die zur Betrachtung vier unterschiedliche Code-Snippets vorgelegt bekamen. Der Code beschreibt einmal in gutem (OK) und einmal in schlechtem (Smell) Stil das Objekt „Rectangle“ sowie das Objekt „Vehicle“.
Eine Gruppe bekam zunächst den in gutem Stil geschriebenen Code „Recktangle-OK“ (RO) und anschließend den in schlechtem Stil geschriebenen Code „Vehicle-Smell“ (VS) vorgelegt.
Der anderen Gruppe wurden die Code-Snippets „Vehicle-OK“ (VO) und „Rectangle-Smell“ (RS) vorgelegt.

Diese Arbeit führt eine deskriptive statistische Analyse der Metriken durch und untersucht mittels Regressionsanalyse, ob ein Zusammenhang zwischen der Art der Betrachtung und dem Stil des Codes erkennbar wird.

2 Datenbasis

Als Datengrundlage kommen Daten in tab-separierten Format zur Anwendung, die im Rahmen der Doktorarbeit (‚Titel', ‚Jahr') von Herrn Dr. Fabian Deitelhoff erarbeitet wurden und für die vorliegende Arbeit in Form mehrerer „.tsv"-Dateien zur Verfügung gestellt wurden.
Die Daten liegen in folgender Ordnerstruktur vor:

-Fixations // Übergeordneter Ordner

- Fixations.tsv // Beispieldatei eines Probanden

- Rectangle-OK-1920-1080 // Ordner mit 28 Probandendateien
  - 1_RO.tsv
  […] - 54_RO.tsv // Dateien mit EyeTracking Daten der Probanden.

- Rectangle-Smell-1920-1080 //Ordner mit 27 Probandendateien
  - 1_RS.tsv
  […] - 55_RS.tsv// Dateien mit EyeTracking Daten der Probanden.

-Vehicle-OK-1920-2080 //Ordner mit 27 Probandendateien
  - 1_VO.tsv
  […] - 55_VO.tsv// Dateien mit EyeTracking Daten der Probanden.

-Vehicle-Smell-1920-1080 //Ordner mit 28 Probandendateien
  - 1_VS.tsv
  […] – 54_VS.tsv

Es wurden 55 Probanden in zwei unterschiedlich großen Gruppen getestet. Ihnen wurde zwei von vier unterschiedlichen Codes zur visuellen Analyse vorgelegt. Die Dateien enthalten unter anderem die für diese Auswertung wichtigen Daten über die Postition der Fixierung eines Probanden (FixationPosX, FixationPosY) sowie die Dauer dieser Fixierung (Fixation Duration).

Zusätzlich wurden die Daten mit der vordefinierten Areas of Interest in Form einer .csv Datei übergeben. Die Daten enthalten für jeden der vier Codes pro Zeile eine Area of Interest (AOI). Des Weiteren wurden die AOI in Regionen zusammengefasst, so dass beispielsweise 6 Zeilen mit je einer AOI eine gemeinsame AOI-Region darstellen.
Diese AOI-Regionen werden für die Berechnungen in dieser Arbeit verwendet um eine „Area of Interest" darzustellen. Insgesamt wurden in jedem Code sechs AOI-Regionen definiert.

3 Metriken der Eye-Tracking Methodik

Die Arbeit von Sharafi et al. (2015) versucht alle im Software Engineering verwendeten Metriken zusammenzufassen und zu gliedern. Prinzipiell wird hier zwischen vier Arten von Metriken unterschieden.

1) Metriken basierend auf der Fixierung (Fixation)
2) Metriken basierend auf den Augenbewegungen (Sakkaden od. Saccades)

3) Metriken basierend auf dem Blickverlauf (Scanpaths)
4) Metriken basierend auf der Pupillengröße (Pupil size) und der Blinzelfrequenz (Blink rate)

Sharafi et al. (2015) beschreibt die Fixierung als Stabilisierung des Auges auf einem Teil eines Stimulus in einem Zeitraum von 200 – 300 ms wobei zwei Annahmen (Unmittelbarkeitsannahme und Eye-Mind Annahme) zu Grunde gelegt werden. Erstere geht davon aus, dass Teilnehmer bei Erkennen eines Wortes versuchen, es zu interpretieren, während die zweite besagt, dass Teilnehmer ihre Aufmerksamkeit auf ein Wort verwenden bis sie es verstehen. Diese Annahmen wurden von Just und Carpenter (1980) aufgestellt.

Die Sakkaden werden von Sharafi et al. (2015) als schnelle und kontinuierliche Augenbewegungen von einer Fixierung zur anderen beschrieben. Die Geschwindigkeit liegt zwischen 40 und 50 ms. Solche Sakkaden treten willentlich auf, wohingegen unfreiwillige Mikrosakkaden während langen Fixierungen ebenfalls vorkommen. Diese dienen der Auffrischung des visuellen Gedächtnisses. (Sharafi et al., 2015)

Der Blickverlauf (Scanpath) ist eine chronologische Reihe von Fixierungen die das Bewegungsmuster der Augen darstellen während die Pupillengröße die Erweiterung der Pupille beschreibt. (Sharafi et al., 2015)

3.1 Allgemeine Metriken

Sharafi et al. (2015) fasst eine Reihe von Metriken in Form einer Tabelle zusammen, die in der folgenden Liste unter dem Oberbegriff der entsprechenden Einordung dargestellt werden und einen Überblick über bestehende Metriken geben sollen:

a) Fixierungen:
   a. *Number of Fixations:*
      Fixation Count (FC), Fixation Rate (FR), Fixation Spatial Density (SD), Convex hull;
   b. *Duration of Fixation:*
      Average Fixation Duration (AFD), Ratio of ON-target: All-target Fixation Time (ROAFT), Fixation Time (FT), Average Duration of Relevant Fixations (ADRF), Normalized Rate of Relevant Fixations (NRRF);
b) Sakkaden:
   Number of Saccades, Saccade Duration, Regression Rate;

c) Blickverlauf (Scanpath):
   Attention Switching Frequency, Transitional Matrix, Edit Distance, Sequential Pattern Mining (SPAM), Scan Match, Linearity;

d) Pupillengröße und Blinzelfrequenz (pupil size and blink rate):
   Blink rate, Pupil size;

3.2 Auswahl der Metriken für die Überführung in Java Objekte

Sharafi et al. (2015) erläutert, dass vorangegangene Studien in der Regel Fixierungsbasierte und Scanpfad-basierte Metriken verwendeten, wohingegen die schnellen Augenbewegungen (Sakkaden) nur selten genutzt wurden. Dies wird damit begründet, dass nach allgemeinem Konsens der Eye-Tracking Forschung kognitive Verarbeitung und Verständnis während der Fixierungen geschieht, die Verarbeitung während der Sakkaden allerdings nur begrenzt stattfindet. Auch Pupillengröße und Blinzelfrequenz wurden kaum verwendet, da diese Größen sehr empfindlich auf das Umgebungslicht reagieren und leicht von mehreren äußeren Einflüssen beeinträchtigt werden können.
Aus diesem Grund kommen für die vorliegende Arbeit ausschließlich Metriken basierend auf der Fixierung zur Anwendung.

Um dem Rahmen der Arbeit gerecht zu werden, werden nur einige der Metriken aus den Fixierungen ausgewählt. Diese sollen den Aufwand bis zum Verstehen eines Betrachtungspunktes näherungsweise darstellen können.

Als Metriken der *Anzahl an Fixierungen* stellen die **Fixation Rate (FR)** und die **Fixation Spatial Density (SD)** relevante Informationen dar.

Ausgehend der Tabelle I in Sharafi et al. (2015) werden die Ergebnisse der Metrik Fixation Rate (FR) mit steigender Rate in Verbindung mit einer höheren Effektivität bzw. einem geringeren Aufwand beim Verständnis des betrachteten Objekts gebracht. Die FR ergibt sich aus dem Verhältnis der Anzahl aller Fixierungen in einem Interessenblickfeld (Area of Interest [AOI]) zu der Anzahl der Fixierungen im gesamten Blickbereich (Area of Glance [AOG]).

$$FR = \frac{Total\ Number\ of\ Fixations\ in\ AOI}{Total\ Number\ of\ Fixations\ in\ AOG} \quad (1)$$

Die Ergebnisse der Fixation Spatial Density (SD) hingegen weisen mit steigendem Wert auf eine direktere Suche, beziehungsweise auf ein geringeres Zeitinvestment (Aufwand) bei der Betrachtung des Objekts hin. Sie entspricht der Division aus der Anzahl der Zellen die mindestens eine Fixierung enthalten ($\sum_{i=1}^{n} c_i$) und der Gesamtanzahl der Zellen wenn der Stimulus als Gitter betrachtet wird (n).

$$SD = \frac{\sum_{i=1}^{n} c_i}{n} \quad (2)$$

Aus den Metriken der *Dauer der Fixierungen* ist die **Average Fixation Duration (AFD) i**nteressant.

Die Average Fixation Duration (AFD) weist nach Tabelle II aus Sharafi et al. (2015) mit steigendem Wert auf eine Zunahme des Aufwands bzw. einen steigenden Schwierigkeitsgrad im Verständnis des betrachteten Objekts hin. Sie lässt sich aus dem Quotienten der Summe der Dauer aller Fixierungen in einer AOI ($\sum_{i=1}^{n}(ET(F_i) - ST(F_i))$) und der Anzahl der Fixierungen (n) berechnen.

$$AFD\ (AOI) = \frac{\sum_{i=1}^{n}(ET(F_i) - ST(F_i))}{n} \quad (3)$$

4 Statistische Methodik

Die Ergebnisse der Eye Tracking Untersuchungen sollen mit Hilfe von Java statistisch ausgewertet werden. Hierzu wird eine deskriptive Statistik unter Verwendung des Moduls org.apache.commons.math3 durchgeführt.
Zusätzlich wird die Häufigkeitsverteilung einer Metrik durch Histogramme dargestellt.
Zur Durchführung der Regressionsanalyse wird das Modul org.deeplearning4j verwendet.

4.1 Deskriptive Statistik

Die deskriptive Statistik soll einen Überblick über die Ausprägungen des Datensatzes darstellen und beschränkt sich daher auf die Erhebung der wichtigsten statistischen Kennwerte. Unter Anderem werden die folgenden Kennwerte berechnet:
Anzahl, Minimum, Maximum, Summe, Arith.Mittelwert, Median, Varianz, und Standardabweichung.
Zusätzlich wird die Häufigkeitsverteilung der Werte einer Metrik in Form von Histogrammen abgebildet.

4.2 Regressionsanalyse

Die Regressionsanalyse soll einen potentiellen Zusammenhang zwischen den Metriken und dem beobachteten Objekt darstellen können.
Hierzu werden die Metriken als unabhängige Variablen definiert und in Abhängigkeit zum beobachteten Ereignis, der Betrachtung eines Codes im schlecht geschriebenen Stil (Code-Smell) und der Betrachtung eines Codes im gut geschriebenen Stil (Code-OK) gesetzt.

Es wird ein Modell erzeugt, das anhand der vorhandenen Daten trainiert und an einem Testdatensatz angewandt wird. Das Ziel ist eine Aussage über die Qualität des Codes dieses Testdatensatzes auf Basis der EyeTracking – Daten zu treffen.
Da die Zielvariable nur zwei mögliche Zustände kennt (Code-OK [0]/ Code-Smell [1]) ist der Ansatz einer binomialen logistischen Regression sinnvoll. Diese hilft dabei, Klassifizierungen anhand der unabhängigen Variablen durchzuführen.

Hierzu wird eine Sigmoid-Funktion (σ) verwendet. Die abhängige Variable lässt sich dadurch mit einer bestimmten Wahrscheinlichkeit zuordnen. Die folgende Formel veranschaulicht die Berechnung. Die Variablen $x_1 – x_4$ stellen die unabhängigen Variablen dar. Die Variable σ stellt die Sigmoid Funktion, $\theta_0 – \theta_4$ die Wichtung der unabhängigen Variablen dar.

$$\hat{y} = \sigma * (\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_4 x_4) \quad (4)$$

Das Ergebnis ŷ ergibt dann einen Wert zwischen 0 und 1 und stellt die Wahrscheinlichkeit P(Y=1|X) dar, mit der die abhängigen Variablen den Zustand 1 (Code-Smell) abbilden. Der Fehler der Vorhersage für ein Element (einen Probanden) wird durch die Subtraktion 1 – P(Y=1|X) errechnet. Die mittlere quadratische Abweichung wird mit Hilfe einer Verlustfunktion berechnet, welche den Fehler des Modells angibt.

Um das Modell einer logistischen Regression möglichst korrekte Aussagen treffen zu lassen, ist es also angebracht, die Summe der Fehler zu minimieren. Dieser Schritt wird in einem

iterativen Prozess durch die Anpassung der θ-Werte vollzogen. Als Hilfsmittel für die Anpassung dient die Gradientenmethode. Hierbei wird der Gradient der Verlustfunktion gebildet um das Minimum herauszufinden. (Grus, 2020)

5 Überführung in Java-Objekte

Um die Daten in Java zu verarbeiten, sind mehrere Schritte notwendig. Da es sich um eine objektorientierte Programmiersprache handelt, können für viele Aufgaben Objekte erstellt werden.
Deren Klasse kann dann mit Methoden ausgestattet werden, die die Handhabung der Daten steuert und erleichtert. Im Folgenden werden die unterschiedlichen Schritte bis zur statistischen Berechnung erläutert. Analog dazu lässt sich der Java-Code in IntelliJ verfolgen.

5.1 Einladen der Daten

Die übergebenen Daten müssen in eine für Java lesbare Form übertragen werden. Hierzu ist es erforderlich, eine Möglichkeit zu erarbeiten, die Probanden-Daten (.tsv-Format) an Java zu übergeben.
Um einen Abgleich der Fixierungen der Probanden (.tsv -Format) mit den AOI-Regionen (.csv-Format) zu ermöglichen, müssen zudem die AOI Regionen aus dem .csv – Format in eine für Java lesbare Form gebracht werden.

Hierzu wurden drei Klassen (AreaOfInteresCSV,AoiCoordinates und LoadingTsv) erstellt.
Die Methode der Klasse „AreaOfInterestCSV“ liest einen angegebenen Orderpfad ein und überprüft, ob hinter dem Pfad eine Datei enthalten ist. Anschließend wird die Klasse „AoiCoordinates“ genutzt, um aus den in der Datei enthaltenen Koordinaten ein Koordinaten-Objekt zu erstellen. Die Ausgabe der Methode *readData* der Klasse „AreaOfInterestCSV“ ist ein HashMap-Objekt, das die Region (A-F) und die entsprechenden vier Koordinatenpunkte dieser Region (AoiCoordinates-Objekt) beinhaltet.

Der Konstruktor der Klasse „LoadingTsv“ liest einen angegebenen Ordnerpfad ein und überprüft, ob hinter dem Pfad eine Datei enthalten ist. Anschließend erzeugt er ein „LoadingTsv“ – Objekt.
Die Methoden des „LoadingTsv“ – Objekts erlauben es, unterschiedliche Parameter aus der Datei zu entnehmen und die Werte dieser Parameter in Variablen und Arrays abzulegen. Dadurch wird ein Zugriff auf die Daten in Java ermöglicht.
Nun genügt es nicht, die Werte in einem Array oder einer Variablen abzulegen, da mehrere Probanden mit unterschiedlichen Messungen und Werten existieren. Es ist also sinnvoll, für die Probanden eine Klasse zu erzeugen.

5.2 Überführen von Daten in Objekte

Um die Daten in Objekte zu überführen, wurden die Klassen „Proband“ und „Probands“ erstellt. Die Klasse „Proband“ nimmt Daten in der Form von Integer Arrays und Strings auf. Mit Hilfe des Konstruktors wird zunächst ein „Proband“ – Objekt erzeugt.
Um das Objekt mit Daten zu befüllen wurde eine *Setter-Methode* erstellt. Die *Setter-Methode* der Klasse Proband fordert als Input die Variablen „name“ (String), „fixationDurations“

(ArrayList<Integer>), „fixationsPosX (ArrayList<Integer>) und „fixationPosY“ (ArrayList<Integer>). Diese Variablen werden mit Hilfe des „LoadingTsv“-Objekts übergeben. Damit wird dann das Objekt der Klasse „Proband“ gefüllt.
Mit Hilfe der *Getter-Methoden* lässt sich dann einzeln auf die hinterlegten Variablen zugreifen. Die Fähigkeit einzelne Variablen zu übergeben, beinhaltet auch die Möglichkeit, eventuell benötigte weitere Variablen nachträglich hinzuzufügen.

Mit der Klasse „Probands“ wird mit Hilfe der Methode *accessProbands* durch den Zugriff auf die statisch festgelegten Dateipfade eine ArrayList aller Probanden in den in der Methode angegebenen Ordner (Rectangle-OK („RO“), Vehicle-Smell („VS“), Rectangle-Smell („RS“) oder Vehicle-OK („VO“)) erzeugt.
Die Methode *accessProbands* gibt dann eine ArrayList aller Probanden aus.

5.3 Erstellen von Klassen für ausgewählte Metriken

Die ausgewählten Metriken aus Kapitel 3.2 sollen auf die vorhandenen Eye-Tracking Daten angewandt werden. Hierzu muss die Berechnungsgrundlage zunächst in einer in Java-lesbarer Art hinterlegt werden.
Dazu wurde für jede Metrik eine Klasse erstellt.
Bei den Klassen „AverageFixationDuration“(AFD) und „und FixationRate“(FR) wird zunächst mittels Konstruktor ein Objekt der Metrik erstellt. Dieses Objekt besitzt die Eigenschaft **aoiRegions** in der die AOI-Regionen der jeweiligen Gruppe (RO,VS,RS oder VO) für die Berechnung hinterlegt werden sollen. Zudem werden die Eigenschaften **valuesAfdForOneProband/valuesFrForOneProband** und **valuesAfdForAllProbands/valuesFrForAllProbands** initiiert. Diese nehmen die Ergebnisse der Berechnung des AFD bzw FR – Wertes in Form eines HashMap – Objekts für einen Probanden und jede AOI-Region (A-F) und im zweiten Fall in Form eines HashMap -Objekts für jeden Probanden einer Gruppe (RO,VS,RS,VO) mit jeder AOI-Region (A-F) auf.

Mittels Konstruktor wird das Objekt initiiert.

Die Methode *calcAfdForOneProband* bzw. *calcFrForOneProband* des Metrik - Objekts beinhaltet die Berechnungsgrundlage der jeweiligen Metrik und fordert als Input ein Probandenobjekt sowie die zugehörigen Gruppenbezeichnung (RO,VS,VO,RS).
Die Ausgabe der Methode *calcAfdForOneProband* bzw. *calcFrForOneProband* des Metrik-Objekts liefert schließlich die berechnete Metrik die in der Eigenschaft **valuesAfdForOneProband/valuesFrForOneProband** abgelegt ist.

Mit Hilfe der Methode *setAfdList* bzw. *setFrList* lässt sich unter Angabe eines Probands - Objekts und der zugehörigen Gruppenbezeichnung (RO,VS,VO,RS) die entsprechende Metrik für jeden Probanden berechnen und im jeweiligen Metrik-Objekt unter der Eigenschaft **valuesAfdForAllProbands/valuesFrForAllProbands** hinterlegen.
Dann kann über die Methode *getAfdList* bzw. *getFrList* auf die Ergebnisse der Metrik für jeden Probanden der zuvor angegebenen Gruppe zugegriffen werden, um weitere Berechnungen durchzuführen. Ausgabe ist dann ein HashMap-Objekt.

Da die Metrik Fixation Spatial Density (FSD) unabhängig von der AOI-Region berechnet wird, ist die Angabe dieser Regionen für die Klasse überflüssig.
Die Klasse „fixationSpatialDensity“ besitzt als Eigenschaft das HashMap-Objekt **valuesFsdForAllProbands.** Sie wird mittels Konstruktor initiiert.
Unter der Angabe eines Probanden-Objekts lässt sich mit Hilfe der Methode *calcFsdForOneProband* die Fixation Spatial Density für einen Probanden berechnen.
Analog zu den anderen Metrik-Objekten lässt sich die FSD für alle Probanden einer Gruppe mit der Methode *setFsdList* und dem Input eines Probands-Objekts berechnen.
Das Ergebnis wird in der Eigenschaft **valuesFsdForAllProbands** hinterlegt. Über die Methode *getFsdList* kann auf die Ergebnisse zugegriffen werden. Ausgabe ist auch hier ein HashMap-Objekt.

5.4 Berechnung der statistischen Kennwerte und Export in Ergebnisdatei

Die statistische Berechnung muss durch eine Überführung der statistischen Methoden in Java Objekte für den Zugriff auf die Daten ebenfalls erst zugänglich gemacht werden.
Für jede statistische Berechnung wurde daher zunächst ein Objekt erstellt. Diese Objekte sind in Paketen strukturiert worden.
Es wurden zwei Pakete mit der Bezeichnung Statistics und StatisticalEvaluation erstellt. Das Paket Statistics beinhaltet die Berechnungsgrundlagen (Deskriptive Statistik, Konfiguration des neuronalen Netzes) für die weitere statistische Auswertung, während das Paket StatisticalEvaluation komplexere Auswertungsmethoden wie die Erstellung der Histogramme sowie die Durchführung der Regressionsanalyse zur Auswahl stellt.

5.4.1 Berechnung der deskriptiven Statistik

Um die deskriptive Statistik zu berechnen, wird zunächst die Klasse „DescriptiveStatisticalAnalysis“ im Paket „Statistics“ erstellt.
Diese Klasse wird mittels Konstruktor initiiert und besitzt zwei Methoden.
Da die Objekte der Metriken „AverageFixationDuration“ und „FixationRate“ ein MapObjekt mit anderem Inhalt ausgeben, als das Metrik-Objekt „FixationSpatialDensity“, werden zwei unterschiedliche Methoden entworfen.

Die Methode *summaryStatsAfdFr* berechnet für jede AOI-Region (A-F) aus der Summe aller Probanden in einer Gruppe (RO, VS, RS oder VO) eine deskriptive Statistik und gibt diese als String aus. Zur Durchführung der Berechnung der statistischen Kennwerte wird das Paket org.apache.commons.math3.stat.descriptive.SummaryStatistics verwendet.

Die Methode *summaryStatsFsd* berechnet die deskriptiven Statistiken für die von der AOI-Region unabhängige Metrik „FixationSpatialDensity“ aus der Summe aller Probanden in einer Gruppe (RO, VS, RS oder VO). Die Kennwerte werden ebenfalls mit dem Paket SummaryStatistics aus dem Modul org.apache.commons.math3 berechnet. Diese Methode gibt nach Aufruf ein „SummarStatistics“ Objekt aus, was den Vorteil bietet, dass sich aus diesem Objekt außerhalb der Klasse „SummaryStatistics“ auf alle berechneten statistischen Kennwerte einzeln zugreifen lässt. So können diese in weiterführenden Analysen verwendet werden.

Zur Auswertung der deskriptiven Statistik der Fixation Spatial Density wurde die Klasse „DescriptiveStatisticsToCSV“ im Paket SavingData erstellt. Diese Klasse besitzt die Methode *createCSVfromFSD* welche als Input für jede Gruppe (RO, VS, RS, VO) Objekte der Klasse „SummaryStatistics“ benötigt sowie den Pfad und den Namen unter dem die in der Methode erzeugte .csv-Datei gespeichert werden soll.

5.4.2 Berechnung der Histogramme

Um eine Darstellung von Histogrammen zu ermöglichen, wurde im Paket StatisticalEvaluation die Klasse „Histogramm“ erstellt. Sie wird mittels Konstruktor initiiert und lässt sich durch die Methode *showHistogram* ausführen. Die Klasse „Histogramm“ dient ausschließlich zur Auswertung der Metrik ‚Fixation Spatial Density‘.
Die Methode fordert als Input das Ergebnis der Methode *getFsdList* aus der Klasse „FixationSpatialDensity“ (HashMap-Objekt) sowie die Angabe der Klassenanzahl in Form eines Integer-Wertes. Innerhalb der Methode wird dann errechnet wie sich die Fixation Spatial Density – Werte (FSD) in einer Gruppe (RO,VS,RS,VO) verteilen. Die Methode liefert als Ergebnis einen String der die Klassengrenzen sortiert nach deren Größe übereinander anzeigt. In der Zeile jeder Klassengrenzen wird die Anzahl der Werte innerhalb dieser Grenzen in Form mehrere * - Symbole verdeutlicht.

5.4.3 Logistische Regressionsanalyse

Zur Durchführung der Regressionsanalyse wurde das Modul org.deeplearning4j verwendet. Es bietet unterschiedliche AI-Methoden für Java an, unter Anderem auch die logistische Regression.
Dazu wird ein „MultiLayerConfiguration“ Objekt erstellt (org.deeplearning, in welchem die Startparameter für die Regressionsanalyse festgelegt werden. Dieses Objekt stellt das Model dar, das anschließend auf den Training-Datensatz angewandt wird und mit dem Test-Datensatz geprüft wird. Auf der Website des Anbieters ist ein Beispiel zur Durchführung der logistischen Regression aufgeführt, an dem sich die in dieser Arbeit durchgeführte Regressionsanalyse orientiert. (dl4j, 2021)

Zur Umsetzung der Regressionsanalyse für die vorliegenden Eye-Tracking Daten wurden drei neue Klassen erzeugt.

Die Klasse „NeuralNetConfigurationSetUp“ wurde im Paket Statistics erstellt. Sie wird mittels Konstruktor initiiert und verwendet die Methode *setUpLogisticRegression.* Diese Methode gibt ein „MultiLayerConfiguration“ – Objekt aus und benötigt als Eingangsparameter die von diesem Objekt verwendeten Eingangsdaten **FEATURES_COUNT** und **CLASSES_COUNT**. Dabei sind die Anzahl der unabhängigen Variablen und die Anzahl der möglichen Varianten der Zielvariable (0,1) gemeint.

Um die relevanten Daten für die Regressionsanalyse speichern zu können, wurde die Klasse „BLRVariablesToCSV“ im Paket SavingData erstellt. Sie wird mittels Konstruktor initiiert und besitzt die Methode *saveData,* welche als Input folgende Parameter benötigt:
fsd (Fixation Spatial Density) als ArrayList mit Double-Werten,
afdMax (maximum Average Fixation Duration) als ArrayList mit Double-Werten,

afdDistr (average deviation to Maximum AverageFixationDuration) als ArrayList mit Double-Werten,
frMax (Maximum Fixation Rate) als ArrayList mit Double-Werten,
frDistr (average deviation to Maximum Fixation Rate) als ArrayList mit Double-Werten,
targetVariable (Zielvariable (CodeSmell [1] – CodeOK [0]) als ArrayList mit Integer-Werten
und die Angabe des Pfades und des Dateinamens unter dem die neue Datei gespeichert werden soll.

Die dritte Klasse „BinomialLogisticRegression“ wurde im Paket StatisticalEvaluation erstellt. Hierin wird die Regressionsanalyse durchgeführt sowie die Daten in eine für die Analyse notwendigen Form gebracht und als .csv-Datei abgespeichert. Auch diese Klasse wird mittels Konstruktor initiiert.

Mit zwei Methoden wird die Funktionalität der Klasse umgesetzt. Die erste Methode *createCSVforBLRegression* verwendet die Klasse „BLRVariablesToCSV“, um eine Datei mit den notwendigen Daten zu erzeugen. Sie benötigt als Input die Angabe zweier Gruppen (RO und VS oder RS und VO) um darüber auf die Daten der Probanden aus den entsprechenden Gruppen zugreifen zu können und die Metriken zu berechnen. Als weiteren Input wird der Pfad und der Dateiname benötigt, unter dem die neue .csv-Datei mit den auszuwertenden Metriken abgelegt werden soll.

In dieser Methode wird zudem aus der Metrik Average Fixation Duration die maximale AFD aller AOI-Regionen eines Probanden berechnet und als erste AFD-Variable (AFD-Max) genutzt. In einem weiteren Schritt wird die durchschnittliche Abweichung der AFD-Werte jeder AOI-Region zum maximalen AFD-Wert berechnet. Je geringer die Abweichung desto gleichmäßiger wurden die AOI-Regionen betrachtet. Die einzelnen AOI-Regionen hier nicht näher berücksichtigt, um dem Rahmen der Arbeit gerecht zu werden.

Auch die Metrik Fixation Rate liegt für jede AOI-Region vor und wird daher für die Regressionsanalyse ebenfalls in zwei Werten dargestellt. Ein Wert stellt die maximale Fixation Rate eines Probanden dar, also den höchsten Wert den eine AOI Region in der Häufigkeit der Betrachtung erzielte. Um die Verteilung der anderen AOI-Regionen abzubilden, wurde die mittlere Abweichung der Fixation Rate jeder AOI-Region zu diesem maximalen FR-Wert berechnet. Je höher der Wert dieser Abweichung ist, desto einseitiger wurde die AOI Region betrachtet, die den maximalen FR-Wert aufweist.
Die berechneten Werte werden anschließende in der .csv-Datei gespeichert.

Die zweite Methode *calcBinomialLogisticRegression* benötigt als Input die Werte für die Anzahl der unabhängigen Variablen, der Anzahl der möglichen Zustände der abhängigen Variable (0,1), die Position der abhängigen Variable in der .csv – Datei (Spaltennummer) sowie den Pfad zum vorher erzeugten Testdatensatz und zum vorher erzeugten Trainingsdatensatz. Diese sind mit der zuvor erwähnten Methode zu erstellen.
Die Daten aus den .csv-Dateien werde in der Methode eingelesen und normiert. Anschließend wird das in der Klasse „NeuralNetworkConfigurationSetUp“ aufgesetzte Modell verwendet und an den Trainingsdatensatz angepasst. Mit Hilfe des Testdatensatzes wird das Modell schließlich ausgewertet.

Die Methode gibt zuletzt die Auswertungsstatistik des auf den Testdatensatz angewendeten Modells aus. Hierzu zählen die Parameter accuracy, precision, recall und f1-score sowie eine „confusion matrix“ wie hoch die Anzahl richtiger und falscher Modellvorhersagen liegt.

5.5 Beschreibung der Main-Methode

Die main-Methode dient der Durchführung der Statistik sowie dem notwendigen Überführen der Daten in lesbare Objekte. Sie ist in drei Teile untergliedert. Der erste Teil dient als Berechnungsprobe der Metriken für einen Einzigen Probanden und wird beispielhaft für zwei Probanden der Gruppe „RO“ durchgeführt.

Im zweiten Teil werden Objekte aller Metriken (AFD, FR, FSD) für alle Probanden, unterschieden nach der Codegruppenzugehörigkeit (RO, VS, RS, VO), durchgeführt. Dadurch ergeben sich 3 x 4 Objekte, für jede Metrik und jede Gruppe eines. Tabelle 1 veranschaulicht die Zusammenstellung:

Tabelle 1: Metrik-Objekte aller Gruppen

	Rectangle – OK (RO)	**Vehicle – Smell (VS)**	**Rectangle – Smell (RS)**	**Vehicle – OK (VO)**
Average Fixation Duration	afdRO	afdVS	afdRS	afdVO
Fixation Rate	frRO	frVS	frRS	frVO
Fixation Spatial Density	fsdRO	fsdVS	fsdRS	fsdVO

Der dritte Teil dient den statistischen Berechnungen. Hier werden zunächst für jedes der oben dargestellten Objekte die deskriptiven Statistiken ausgegeben. Da die Average Fixation Duration sowie die Fixation Rate für jede AOI-Region berechnet wurden, liegt die Deskriptive Statistik auch für jede AOI-Region vor. Für die deskriptiven Statistiken der Fixation Spatial Density wurde

Auf die deskriptive Statistik folgt die Auswertung der Binomialen Logistischen Regression. Hierbei wird zunächst der Trainingsdatensatz erzeugt. Wie in Kapitel 5.4.3 erwähnt, ist es notwendig, zur Durchführung der Methode *createCSVforBLRegression* den Speicherpfad und den gewünschten Namen der Datei anzugeben. Für den Trainingsdatensatz wird der Name TrainData.csv verwendet.
Des Weiteren müssen die für diesen Datensatz gewünschten Codegruppen angegeben werden. Für den Trainingsdatensatz wurden die Gruppen RO und RS verwendet, da hier unterschiedliche Probanden den Code gelesen haben. Analog dazu wurde der Testdatensatz mit den Gruppen VS und VO unter dem Namen TestData.csv erstellt.
Im nächsten Schritt wird dann die Binomiale Logistische Regression durchgeführt.

Zuletzt werden die Histogramme der Metrik Fixation Spatial Density für jede Codegruppe (RO, VS, RS, VO) dargestellt.

6 Ergebnisse der statistischen Auswertung

Die relevanten Kennwerte der Ergebnisse der deskriptiven Statistik der Fixation Spatial Density sind in Abbildung 1 veranschaulicht.

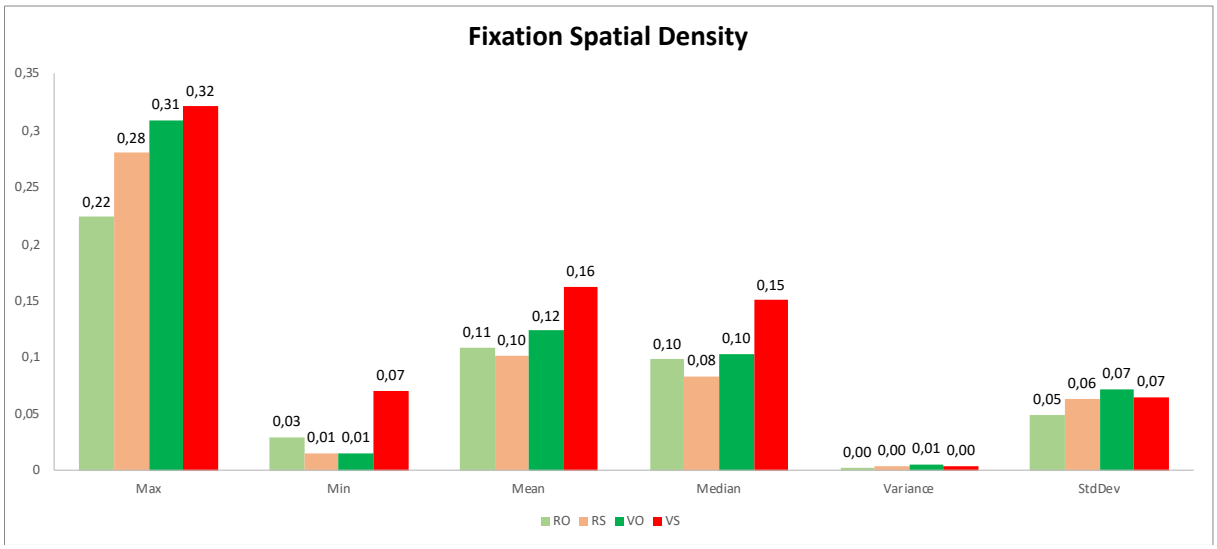

Abbildung 1: Darstellung statistischer Kennwerte der FixationSpatialDensity aller Gruppen

Die maximalen FSD-Werte unterscheiden sich zwischen gutem und schlechtem Code. Die Maxima des schlecht geschriebenen Codes (VS,RS) liegen um 0.1 bzw. 0.5 höher als die des gut geschrieben Codes (VO,RO).
Bei den anderen Kennwerten sind keine gemeinsamen Besonderheiten erkenntlich.
Die Minima zwischen RS und VO liegen auf etwa gleichem Niveau während sie bei den Gruppen RO und VS einen Unterschied von 0.5 aufweisen. Der FSD-Wert des schlecht geschrieben Codes des Objekts Rectangle liegt im Minimum unter dem Wert des gut geschrieben Codes des Objekts Rectangle, wohingegen sich dieses Verhältnis bei dem Objekt Vehicle umkehrt und das Minimum des gut geschrieben Codes unter dem Minimum des schlecht geschriebenen Codes liegt.
Auch beim Mittelwert, dem Median, der Varianz und der Standardabweichung sind diese Verhältnisse umgekehrt.
Allerdings zeigt sich, dass die FSD allgemein etwas größere Werte bei dem Objekt Vehicle annimmt.

Eine weiter Auffälligkeit liegt darin, dass die Spanne zwischen den arithmetischen Mittelwerten der FSD-Werte bei schlecht geschriebenem Code (0.06) sehr viel größer ist, als bei gut geschriebenem Code (0.01). Diese Beobachtung trifft auch für den Median zu (Smell: 0.07, OK: 0.006)

Um den Rahmen der Arbeit nicht zu überdehnen, wird auf die deskriptive Statistik der Metriken Fixation Rate (FR) und Average Fixation Duration (AFD) nur kurz eingegangen. Am auffälligsten zeichnet sich hierbei ab, dass von den 7 AOI-Regionen A-F insbesondere die Region F keine Werte aufweist. Es ließen sich also keine FR/AFD Werte errechnen, weil von allen Probanden unzureichend Fixierungen in dieser AOI Region vorgenommen wurden.
Die Unterschiede der FR- und AFD – Werte zwischen den AOI Regionen werden im Rahmen der Regressionsanalyse berücksichtigt und durch die Verwendung der maximalen Werte der Metrik eines Probanden und der mittleren Abweichung zu diesem Maximalwert in die Rechnung miteinbezogen.

Für die Metrik Fixation Spatial Density wurden Histogramme erzeugt. Jedes Histogramm stellt die Verteilung der FSD-Werte innerhalb einer Codegruppe (RO, VS, RS, VO) dar. Auffallend ist,

dass alle Histogramme rechtsschiefe Verteilungen aufweisen. Der FSD-Werte liegen also bei allen Codegruppen zu einem größeren Teil in niedrigeren bis mittleren Wertebereichen.
Zudem lässt sich feststellen, dass die Verteilung der FSD-Werte der Codegruppen RO und VS ähnlich ausgeprägt ist. Beide weisen einen größeren Anteil von Werten im höheren Wertebereich auf, wie Abbildung 2 veranschaulicht.

FSD RO
----------Histogramm Plot----------
[0.0294, 0.0682]: *******
[0.0682, 0.107]: ********
[0.107, 0.1457]: ******
[0.1457, 0.1845]: *****
[0.1845, 0.2233]: **

FSD VS
----------Histogramm Plot----------
[0.0699, 0.1201]: ********
[0.1201, 0.1702]: *********
[0.1702, 0.2204]: ****
[0.2204, 0.2705]: *****
[0.2705, 0.3207]: **

Abbildung 2: Histogramme der FSD-Werte der Gruppen RO und VS

Analog dazu lässt sich die FSD -Verteilung für die Codegruppen VO und RS betrachten. Die Gemeinsamkeit bei diesen Gruppen liegt in einem niedrigeren Anteil von Werten im höherem Wertebereich. Abbildung 3 zeigt die Verteilung der FSD Werte.

FSD VO
----------Histogramm Plot----------
[0.0149, 0.0737]: *******
[0.0737, 0.1324]: **********
[0.1324, 0.1912]: *******
[0.1912, 0.2499]: *
[0.2499, 0.3086]: *

FSD RS
----------Histogramm Plot----------
[0.0149, 0.0679]: ********
[0.0679, 0.1209]: **********
[0.1209, 0.1738]: ****
[0.1738, 0.2268]: *
[0.2268, 0.2797]: *

Abbildung 3: Histogramme der FSD-Werte der Gruppen VO und RS

Die Binomiale Logistische Regression wird auf den Testdatensatz mit den Codegruppen VO und VS angewendet, nachdem das Modell an den Trainingsdatensatz mit den Codegruppen RO und RS angepasst wird. Auf Grund der unterschiedlichen Probandengruppen beim Lesen

des Codes kann so eine größere Repräsentanz erreicht und die Abhängigkeit von den Probanden im Modell minimiert werden.

Die Ergebnisse der Modell-Vorhersage werden in einer Matrix ausgegeben (Abb. 4) und zeigen, dass für 53 Probanden in 30 Fällen der richtige Code und in 23 Fällen der falsche Code vom Modell ausgegeben wurde. In 11 Fällen wurde „Code-OK“ [0] ausgegeben, wo korreterweise „Code-Smell“ [1] vorlag. In 15 Fällen wurde „Code-Smell“ richtig ausgegeben.
In 12 Fällen wurde „Code-Smell“ ausgegeben, wo eigentlich „Code-OK“ vorlag. In 15 Fällen wurde „Code-OK“ richtig ausgegeben.

```
========================Confusion Matrix========================
 0  1
-------
 15 12 | 0 = 0
 11 15 | 1 = 1

Confusion matrix format: Actual (rowClass) predicted as (columnClass) N times
================================================================
```

Abbildung 4: Confusion Matrix BLRegression VO+VS

Um die Qualität der Ausgabe zu beschreiben, wird zusätzlich eine Ausgabe von Evaluierungsmetriken ausgegeben. Diese beziehen sich im vorliegenden Fall auf die Vorhersage des Falls [1], also „Code-Smell“. Die Werte gehen aus Abbildung 5 hervor.

```
========================Evaluation Metrics========================
 # of classes:    2
 Accuracy:       0,5660
 Precision:      0,5556
 Recall:         0,5769
 F1 Score:       0,5660
Precision, recall & F1: reported for positive class (class 1 - "1") only
```

Abbildung 5: Evaluation Metrics BLRegression VO+VS

Die “Accuracy” beschreibt, wie viele Vorhersagen richtig getroffen wurden. Im vorliegenden Fall 56,6 %.

Die „Precision“ ist ein Maß für die echt-positiven Vorhersagen eines Modells. Sie errechnet sich aus dem Verhältnis zwischen den echt-positiven Vorhersagen („1“ korrekt) und der Anzahl der gesamten korrekten Vorhersagen („1“ und „0“ korrekt). Im vorliegenden Fall nimmt die „Precision“ einen Wert von 55,56 % an.

Der „Recall“ ist das Verhältnis zwischen echt positiven Vorhersagen („1“ korrekt) und der Anzahl der tatsächlich positiven Ereignisse (alle „1“-Werte im Original-Datensatz). Er wird auch „Hit-Rate“ genannt und beschreibt die Trefferquote bei Vorhersagen eines bestimmten Ereignisses. Der Wert für den Recall liegt hier bei 57,69 %.

Der F1 Score fasst die Metriken „Precision“ und „Recall“ zusammen und wird aus dem harmonischen Mittel errechnet. Er nimmt einen Wert von 56,6 % an.

7 Diskussion

Die Verteilung der statistischen Kennwerte der Fixation Spatial Density zeigt, dass Gemeinsamkeiten besonders zwischen den Formen, welche der Code beschreibt (Rectangle, Vehicle), auftreten, wohingegen mit Ausnahme der Maxima keine Unterschiede zwischen der Qualität des Codes (OK, Smell) aus den Kennwerten abzuleiten ist.
Dieser Umstand deutet darauf hin, dass die Form, die der Code beschreibt, einen größeren Einfluss auf die Art der Betrachtung nimmt als der Schreibstil.

Die Histogramme der Fixation Spatial Density weisen insbesondere innerhalb der Codegruppen mit gleichen Probanden Ähnlichkeiten auf. Diese Verteilungsähnlichkeiten deuten darauf hin, dass insbesondere innerhalb einer Probandengruppe (Code RO und VS - Probandengruppe 1; Code RS und VO - Probandengruppe 2) Ähnlichkeiten vorliegen. Die Beobachtung trifft zu, da es sich bei den Probanden, die den Code RO betrachtet haben, um dieselben Probanden handelt, die auch den Code VS gesehen haben.
Diese Ähnlichkeiten lassen darauf schließen, dass die Probanden einen stärkeren Einfluss auf die Werteverteilung haben als die Qualität des Codes.

Die Ergebnisse der binomialen logistischen Regression weisen nur eine schwache Vorhersagequalität auf. Fast die Hälfte der Vorhersagen des Modells wurden falsch getroffen. Die Anzahl echt-positiver und echt-negativer Ergebnisse liegt daher leicht über der Anzahl falscher Vorhersagen.

Aufgrund der Ergebnisse der deskriptiven Statistiken ist zu vermuten, dass in den vorliegenden Stichproben die dem Probanden inhärenten Eigenschaften sowie die Form, welche der Code beschreibt, einen stärkeren Einfluss auf die Fixierungen haben als die Art der Schreibweise des Codes. Durch eine Vergrößerung des Probandenpools und die Einbeziehung weiterer Metriken ließe sich dieser Einfluss unter Umständen herausnivellieren.

8 Literatur

Pramodini et al. (2017): P.A. Punde, M. E. Jadhav and R. R. Manza, "A study of eye tracking technology and its applications," 2017 1st International Conference on Intelligent Systems and Information Management (ICISIM), Aurangabad, India, 2017, pp. 86-90;doi:10.1109/ICISIM.2017.8122153; URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8122153&isnumber=8122132

Sharafi et al. (2015): Sharafi, Zohreh & Shaffer, Timothy & Sharif, Bonita & Guéhéneuc, Yann-Gaël. (2015). Eye-Tracking Metrics in Software Engineering. 10.1109/APSEC.2015.53.

Just und Carpenter (1980): M. A. Just and P. A. Carpenter, “A theory of reading: from eye fixations to comprehension.” *Psychological review*, vol. 87, no. 4, p. 329, 1980.

Grus (2020): Joel Grus (2020). Einführung in Data Science – Grundprinzipien der Datenanalyse mit Python. OReilly, 2.Auflage. dpunkt.verlag GmbH, 69123 Heidelberg.

dl4j (2021): https://deeplearning4j.konduit.ai/getting-started/tutorials/logistic-regression, letzter Aufruf: 30.04.2021