Altersbestimmung anhand von Gangmerkmalen

Einführung

Diese Arbeit beschäftigt sich mit dem Vorhersagen der Altersgruppe einer Testperson aus einer vorher durchgeführten Ganganalyse.

Methoden

Datenbeschreibung

Die Daten wurden von den lehrenden der Veranstaltung „Data Science“ zum Download zur Verfügung gestellt. Sie stammen aus verschiedenen Untersuchungen der Ganganalyse mit dem InvestiGAIT-System. Es wurden 3D-Beschleunigung und 3D-Winkelgeschwindigkeit mit Hilfe von zwei beziehungsweise vier Inertialsensoren erfasst. Relevante Gangereignisse wurden aus den Signalen extrahiert und zu Berechnung verschiedener Gangparameter genutzt. 25 Studenten oder Mitarbeiter der Abteilung Sportwissenschaft der Otto von Guericke Universität Magdeburg nahmen an der Studie teil. Die Versuchspersonen liefen eine Strecke von 15 Metern, diese wurde jeweils mindestens zehnmal wiederholt (Orlowski, et al., 2017). Es wurden zwei Datensätze bereitgestellt. Der Datensatz „measures.csv“ enthält 1285 Samples. Ein Sample besteht aus 94 Features, die die extrahierten Gangereignisse und berechnete Gangparameter darstellen, einer anonymisierten Personenkennung, dem Geschlecht und der vorherzusagenden Altersklasse. Der Datensatz „to_predict.csv“ enthält 333 Samples und ist gleich aufgebaut, es fehlen jedoch die Spalten „P-KennungAnonym“ und „P-Altersklasse“.

Explorative Analyse

Beide Datensätze enthalten fehlende Werte in den Spalten der Oberkörper-Parameter (measures.csv 379, to_predict.csv 109). Diese werden für die Modellierung durch den Mittelwert des jeweiligen Features ersetzt. Doppelte Spalten sind nicht vorhanden. Die Spalten „P-KennungAnonym“ und „P-Altersklasse“ werden vor der Modellierung entfernt. Visualisierungen der Features mittels Histogrammen zeigen für viele eine ungefähre und oft verschobene Normalverteilung. Untersuchungen von Mittelwert, Minimum und Maximum zeigen viele unterschiedlich große Skalen. Daher wird eine Reskalierung in das Intervall [0,1] durchgeführt.

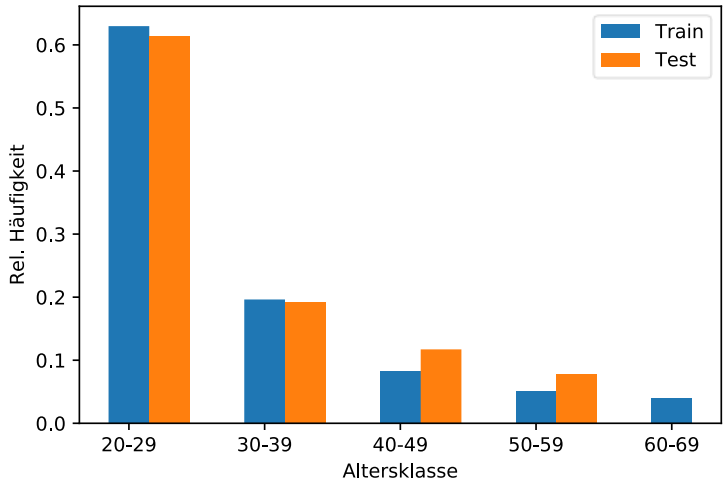

Abbildung 1: Relative Klassenhäufigkeiten in Trainings- und Testmenge, die Klasse "60-69" ist in der Testmenge nicht vorhanden

Measures.csv wird zur Modellierung in Trainings- und Testmenge aufgeteilt, als Testmenge dienen die Personen 3, 26, 27, 32, 37, 44, 47, 51, 54, 57, 60, 61, 63, 66, 70, 90, 95, 109, 111, 116, 120, 123, 134, 149 und 151. Die Klassenverteilung in Trainings- und Testmenge ist annähernd gleich. Jedoch fehlt die Klasse „60-69“ komplett in der Testmenge. Abbildung 1 zeigt die relativen Klassenhäufigkeiten.

Untersuchungen der Korrelationskoeffizienten mit der Klasse zeigen, dass kein Feature sehr stark mit der Klasse korreliert. Die stärksten Korrelationen haben die Features „L-lastStep“ (0.463) und „R-abrollwinkel“ (-0.457). Abbildung 2 zeigt die Boxplots beider Features über der Klasse. Zur Modellierung werden 10 oder 20 Features ausgewählt, jeweils die Hälfte davon aufgrund der höchsten positiven beziehungsweise negativen Korrelation mit der Klasse. Tabelle 1 zeigt die ausgewählten Features und deren Korrelation mit der Klassenspalte.

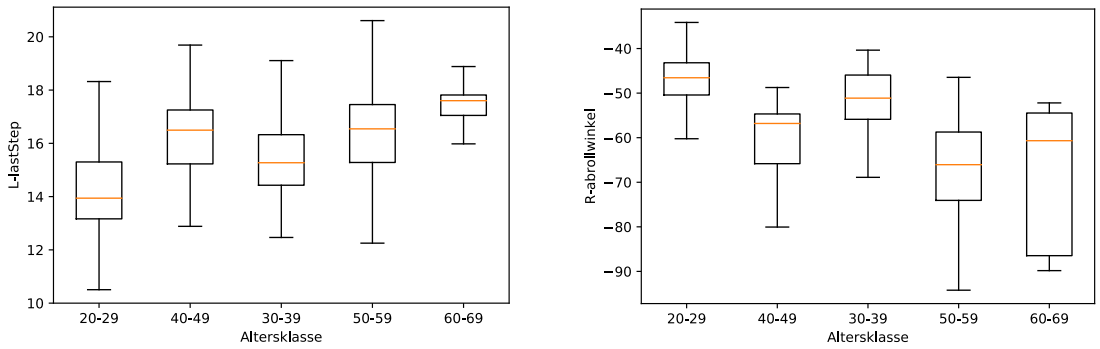

Abbildung 2: Links: Boxplot L-lastStep:Klassen „20-29“ und „60-69“ sind deutlich unterscheidbar voneinander, Rechts: Boxplot R-abrollwinkel:Klasse „30-39“ gut unterscheidbar von allen Klassen außer „20-29“

Feature	Korrelation
L-lastStep	0.462877
R-lastStep	0.436749
L-meanROMHipZ	0.398527
R-meanROMHipZ	0.397658

R-meanAmplMidSwing	0.387617
L-meanAmplMidSwing	0.368202
R-amplDiffTCIC	0.334520
L-amplDiffTCIC	0.333853
R-meanROMSpineZ	0.328727
L-aufsetzwinkel	0.328051
R-abrollwinkel	-0.457236
L-abrollwinkel	-0.395562
S-aufsetzwinkel	-0.291471
S-meanAmplMidSwing	-0.282434
L-meanStrideLen	-0.275388
R-meanStepHeight	-0.264201
S-abrollwinkel	-0.260975
L-velocity	-0.260466
R-velocity	-0.260466
L-velocity2	-0.260466

Tabelle 1: Features mit höchsten Korrelationskoeffizeinten zur Klasse

Modellierung

Es werden Support-Vektor-Maschinen und Multi-Layer-Perzeptrons mit unterschiedlichen Parameter-Einstellungen erprobt. Zur Evaluierung wird eine 5-fache Kreuzvalidierung der Trainingmenge verwendet. Die CV-Splits werden manuell erstellt, da stratifizierte Kreuzvalidierung zu Data-Leakage führen würde (Kaufmann, Rosset, & Perlich, 2011). Zur Partitionierung wird die Personenkennung verwendet. Auf eine gleichmäßige Klassenverteilung in CV-Trainings- und Testmenge wird keine Rücksicht genommen. Als Fehlermaß wird die einfache Klassifikationsgenauigkeit verwendet (Anzahl falsch klassifiziert/Gesamtzahl). In jedem Modelltest werden CV-Resubstitutions- und Testfehler erfasst. Zusätzlich wird ein 80-Prozent-Modell erstellt und auf die Testmenge angewandt. Resubstitutions- und Testfehler des 80-Prozent-Modells werden erfasst. Die Modelle wurden mit scikit-learn erstellt und trainiert (Maschinenausstattung: Intel Core i7 7700k @ 4,20GHz (8 CPUs), 32GB RAM). Die trainierten Modelle sind in Tabelle 2 zu sehen.

Modell	Einstellungen	CV_Resubstitutions-Accuracy	CV_Test-Accuracy	Zeit CV Modell (s)	Resubstitutions-Accuracy 80% Modell	Test-Accuracy 80% Modell	Zeit 80 % Modell (s)
SVM	C = 1.0, kernel = poly, degree = 2, iterations = 100	0.263046 +- 0.080540	0.187316 +- 0.102249	0.229387	0.21671525753158405	0.2265625	0.010971
SVM	C = 1.0, kernel = poly, degree = 2, iterations = 200	0.658201 +- 0.019592	0.608752 +- 0.061601	0.245343	0.652089407191448	0.58984375	0.013963
SVM	C = 1.0, kernel = poly, degree = 3, iterations = 200	0.651366 +- 0.024116	0.608752 +- 0.061601	0.243853	0.6472303206997084	0.58984375	0.015957
SVM	C = 1.0, kernel = poly, degree = 3, iterations = 300	0.651127 +- 0.023893	0.608752 +- 0.061601	0.240863	0.6491739552964043	0.6015625	0.013962
SVM	C = 1.0, kernel = poly, degree = 4, iterations = 300	0.643156 +- 0.019634	0.618176 +- 0.052361	0.249334	0.6394557823129252	0.61328125	0.01496
SVM	C = 1.0, kernel = poly, degree = 5, iterations = 300	0.637160 +- 0.016705	0.619223 +- 0.051657	0.249333	0.6375121477162293	0.61328125	0.013962
SVM	C = 1.0, kernel = rbf, iterations = 100	0.283386 +- 0.111175	0.184027 +- 0.110270	0.268587	0.29640427599611274	0.22265625	0.015957
SVM	C = 1.0, kernel = rbf, iterations = 300	0.679272 +- 0.024203	0.603825 +- 0.062219	0.286234	0.6647230320699709	0.578125	0.023937
SVM	C = 1.0, kernel = rbf, iterations = 400	0.679272 +- 0.024203	0.603825 +- 0.062219	0.279253	0.6647230320699709	0.578125	0.021942
SVM	C = 1.0, kernel = rbf, iterations = 200	0.679272 +- 0.024203	0.603825 +- 0.062219	0.286234	0.6647230320699709	0.578125	0.020944
SVM	C = 2.0, kernel = rbf, iterations = 200	0.727084 +- 0.030817	0.627994 +- 0.081577	0.287232	0.7269193391642371	0.5625	0.020943
SVM	C = 0.5, kernel = rbf, iterations = 200	0.660409 +- 0.018043	0.603825 +- 0.062219	0.290223	0.6491739552964043	0.59765625	0.021941
MLPRegressor	activation = logistic, learning_rate_init = 0.3, solver = adam, iterations = 100	0.629662 +- 0.013133	0.628647 +- 0.049039	0.289226	0.6297376093294461	0.61328125	0.015957
MLPRegressor	activation = relu, learning_rate_init = 0.3, solver = adam, iterations = 100	0.678468 +- 0.025330	0.600470 +- 0.062811	0.316154	0.6822157434402333	0.57421875	0.026928
MLPRegressor	activation = relu, learning_rate_init = 0.1, solver = adam, iterations = 100	0.718973 +- 0.058768	0.586845 +- 0.118080	0.383712	0.7327502429543246	0.59765625	0.042886
MLPRegressor	activation = relu, learning_rate_init = 0.1, solver = adam, iterations = 200	0.713363 +- 0.044317	0.604168 +- 0.070329	0.37501	0.6831875607385811	0.57421875	0.056848
MLPRegressor	activation = relu, learning_rate_init = 0.01, solver = adam, iterations = 200	0.746370 +- 0.059101	0.614354 +- 0.063102	0.568012	0.7968901846452867	0.6484375	0.163562
MLPRegressor	activation = relu, learning_rate_init = 0.01, solver = adam, iterations = 300	0.759261 +- 0.045318	0.611670 +- 0.061092	0.598935	0.760932944606414	0.58203125	0.098854

Tabelle 2:Erprobte Modelle mit Fehlern. Beste Modelle sind grün markiert.

Die drei besten Modelle haben dieselbe Test-Accuracy im 80-Prozent-Modell. Diese unterscheidet sich kaum vom CV- Test-Accuracy. Diese ist beim MLP-Klassifizierer minimal besser als bei den anderen beiden Modellen. Das deutet auf ein minimal stabileres Modell hin.

Abbildung 3 zeigt die Konfusionsmatrix dieses 80-Prozent-Klassifizierers. Die Klasse „20-29“ wird am besten vorhergesagt, jedoch werden die anderen Klassen stark mit dieser verwechselt. Die Klassen „40-49“ und „50-59“ können überhaupt nicht vorhergesagt werden.

Die Analysen wurden in Python 3.6 mit Hilfe der scikit-learn-Bibliothek durchgeführt. Die Skripte können nach Anpassung der Daten ausgeführt werden. Dazu müssen in den CSV-Dateien Kommata durch Punkte ersetzt werden. Durch Ausführen der Skripte und Anpassen der angegebenen Parameter können die Ergebnisse reproduziert werden. Diese können leichte Abweichungen enthalten, da die Modelle mit einem Zufallswert initialisiert werden, der nicht extrahiert oder vorgegeben wurde.

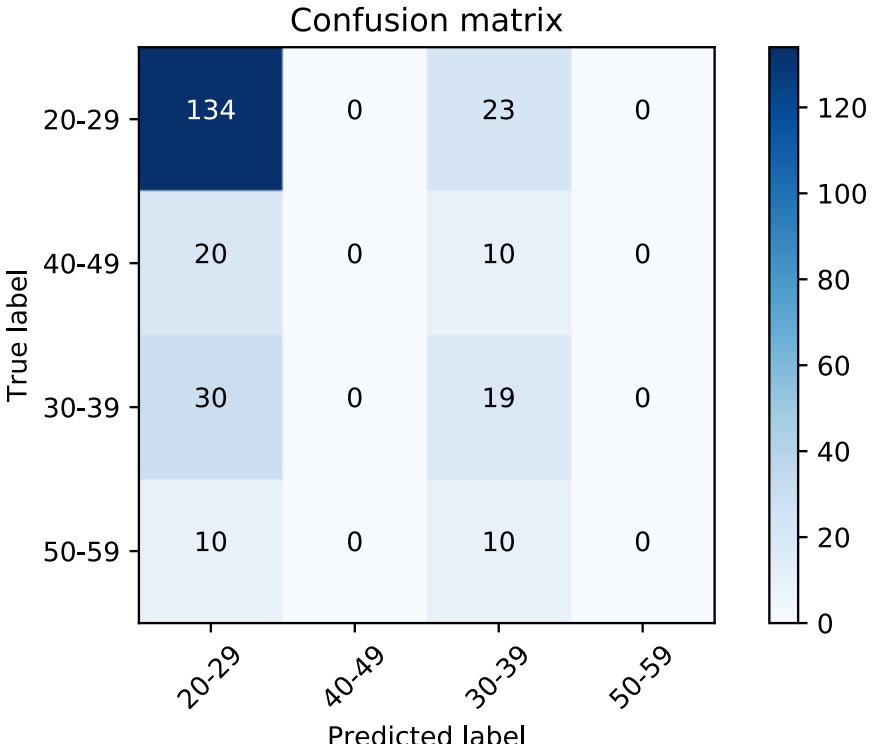

Abbildung 3: Konfusionsmatrix MLP, die Klasse "20-29" wird gut vorhergesagt, jedoch werden alle anderen Klassen mit dieser verwechselt

Ergebnisse

Es wurden Modelle mit 10 beziehungsweise 20 ausgewählten Features erstellt. Modelle mit 20 Features erzielten bessere Ergebnisse, daher wurde auf die Modelle mit 10 Features nicht näher eingegangen. Eine weitere Dimensionsreduktion mittels PCA wurde nicht durchgeführt. Die ausgewählten Features zeigen untereinander teils starke Korrelationen auf (zum Beispiel L-velocity und L-velocity² mit ccf = 1). Korrelierte Signale wurden nicht entfernt, da diese vorher anhand der stärksten Korrelation mit der Klasse ausgewählt wurden. Das Entfernen hätte negative Auswirkungen auf die Modellgüte haben können. Dies wurde jedoch nicht weiter evaluiert. Als Modelltypen wurden Support-Vektor-Maschinen und Multi-Layer-Perzeptrons mit unterschiedlichen Parametern getestet. Beim Training fiel auf, dass MLP mit einer ReLU-Aktivierung schlechter performen als MLP mit logistischer Aktivierung. Beide Modelltypen erzielten als bestes Ergebnis eine Accuracy von 61.33%. Dies ist ein ziemlich schlechtes Ergebnis. Als finales Modell wird ein MLP erstellt, da dieses in der CV-Test-Accuracy minimal bessere Ergebnisse lieferte, was auf ein minimal stabileres Modell hinweist. Dieses Modell hat eine geschätzte Accuracy von 61,33%.

Zusammenfassung

Unsere Analyse zeigt, dass das Vorhersagen der Altersgruppe anhand von Gangmerkmalen möglich ist, aber schlechte Ergebnisse anzunehmen sind. SVM mit polynomialem Kernel performen genauso gut oder minimal schlechter wie MLP mit logistischer Aktivierung und dem Adam Optimierer. Es gibt keinen großen Fehlerunterschied zwischen geschätztem und realem Testfehler. Dies deutet darauf hin, dass die Daten nicht ausreichen, um gute Modelle zu erstellen. Die Auswirkungen einer PCA auf Testfehler wurden nicht untersucht, ebenso wenig wie mehr beziehungsweise weniger Features in der Trainingsmenge. Eine Untersuchung, die beides berücksichtigt, könnte besser Ergebnisse liefern.

Literaturverzeichnis

Kaufmann, S., Rosset, S., & Perlich, C. (2011). Leakage in Data Mining: Formulation, Detection, and Avoidance. *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, S. 556-563.

Orlowski, K., Eckardt, F., Herold, F., Aye, N., Edelmann-Nusser, J., & Witte, K. (2017). Examination of the reliability of an inertial. *Biomedical Engineering / Biomedizinische Technik, 62*(6), S. 615-622.