Projektarbeit

Certified Data Scientist

Vorhersage der Überfallwahrscheinlichkeit in LA auf Basis von LAPD Police Reports

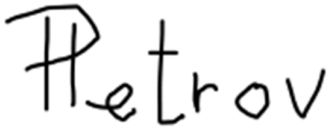
verfasst im Rahmen der EN ISO / IEC 17024-Zertifizierungsprüfung von

Petar Petrov

18.05.2022

Eidesstattliche Erklärung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Projektarbeit eigenständig und ohne Mitwirkung Dritter angefertigt habe. Quellenangaben wurden entsprechend als solche gekennzeichnet.

München, 18.05.2022, 

_____

Ort, Datum, Unterschrift

Inhalt

1 Einleitung

Wikipedia bezeichnet Data Science als „Extraktion von Datenwissen aus Daten“ (Wikipedia, 2022). Diese Daten liegen in sehr großen Mengen vor, woraus auch der Begriff Big Data entsteht. Da diese Datenmengen multidimensional und komplex sind, ist deren Auswertung und Deutung anhand deren Zusammenfassung in Tabellen und Diagrammen unmöglich. Gerade bei mehreren Dimensionen stößt man an die Grenzen des menschlichen Gehirns und Auffassungsfähigkeit. Um diese Aufgabe zu schaffen, werden „Methoden und Vokabular aus den Fachbereichen Statistik, Mathematik, Informationswissenschaften, Computer Science“ eingesetzt (Koch, 2022). Man spricht von Machine Learning Modellen, die in der Lage sind, Muster in großen Datenmenden zu erkennen und diese zu abstrahieren. Dazu wird eine entsprechende „Computational Power“ benötigt. Gerade die technische Entwicklung der Computertechnologie in neuster Zeit führt zu der Ermöglichung der Datenanalyse mit komplexen Modellen aus der Statistik, Mathematik oder Naturwissenschaften. Aufgrund der Verfügbarkeit großer Datenmenden und Rechenleistung sind Unternehmen fast aller Branchen bestrebt, diese zu nutzen, um Wettbewerbsvorteile zu erzielen (Provost, 2015). Ziel der Datenanalyse und damit ihre Unterscheidung zu der klassischen Statistik ist die Vorhersage von Zukunftsereignissen. Diese Prädiktion basiert auf eine vorhandene und relevante Datenbasis und kann eine Aussage über unbekannte Zustände und Entwicklungen in der Zukunft Aussage treffen.

Die Methoden der Datenanalyse finden immer breiteren Einsatz. Die Erkennung von Mustern, Verhaltenstypen oder Targetgruppen ist eine wichtige Aufgabe der Polizei (sog. „Criminal Profiling). Somit ist es nahelegend, dass für die präventive Bekämpfung von Kriminalität auch Datenanalyse-Methoden eingesetzt werden können.

In einem Artikel vom 03.07.2019 in der Online-Ausgabe der LA-Times (www.latimes.com, 2019) beschreibt der Autor den Einsatz der Datenanalysemethoden zu Bekämpfung von Kriminalität wie folgt:

> „The Los Angeles Police Department took a revolutionary leap in 2010 when it became one of the first to employ data technology and information about past crimes to predict future unlawful activity. Other departments around the nation soon adopted predictive policing techniques.”

Die meisten Verbrechen zeichnen sich durch ähnliche Eigenschaften aus. Features wie Motiv (Geld, Aggression, sexuelle Absichten usw.), Waffe, Opfertyp (Frauen, alte Menschen, Kinder), Tätigkeitsgebiet oder spezifische Räumlichkeit (Parkplätze, Eigentumswohnungen, Parks usw.) können in verschiedenen Kombinationen unterschiedlichen Tätern und Opfern zugewiesen werden. Dies würde es der Polizei ermöglichen prädiktiv zu handeln und gezielt Personen und Stadtteile zu beobachten und untersuchen. Eine gezielt verstärkte Polizeipräsenz in kritischen Gebieten oder bei kritischen Ereignissen würde im Gegensatz zu flächendeckenden Präsenz die Kriminalität durch einen smarten Ressourceneinsatz entsprechen reduzieren.

Als Hauptaufgabe der vorliegenden Arbeit wird die Analyse der Kriminalberichte in Los Angeles City festgehalten. Die Programmierumgebung dafür ist Python. Ziel ist es dabei eine Vorhersage der Möglichkeit, dass eine Person angegriffen wird. Es soll weiterhin eine Aussage über Eigenschaften der Straftat wie z.B. Tatort, Tageszeit, bewaffnet oder nicht oder Räumlichkeit getroffen werden. Die Analyse soll für Besucher der Stadt und Touristen eine Hilfe darstellen, Gefahren zu vermeiden.

2 Der Datensatz

Die Internetseite „LOS ANGELES OPEN DATA“ (data.lacity.org, 2022) stellt öffentlich verschiedene Daten in Form von Big-Data-Sets als Information für die Stadt und das Leben der Menschen. Der für diese Arbeit verwendete Datensatz „Crime Data from 2010 to 2019“ stellt eine Sammlung aller gemeldeten Verbrechen in Los Angeles für den Zeitraum zwischen 2010 bis 2019 dar. Die Quelle der Daten ist offiziell die Los Angeles Police Department.

2.1 Die Datenstruktur

Im Datensatz sind über 2 Millionen Einträge dokumentiert. Diese sind strukturiert in 28 Features in Spalten. Die Attribute beziehen sich einmal zum Opfer oder beschreiben das Verbrechen. Anbei eine kurze Beschreibung der für die nachfolgende Datenanalyse wichtigsten Features.

2.1.1 DATE OCC

Datum des Verbrechens in der Form MM/DD/YYYY.

2.1.2 TIME OCC

Uhrzeit des Verbrechens in der Form 24H-military time.

2.1.3 AREA

Die LAPD hat 21 Polizeireviergebiete, die hier nummeriert sind.

2.1.4 Crm Cd Desc

Verbale Beschreibung der Verbrechens nach offiziellen Vorgaben der U. S. Department of Justice.

2.1.5 Vict Age

Alter des Opfers als Zahl

2.1.6 Vict Sex

Geschlecht des Opfers:

- F – female
- M – male
- X – Unknown

2.1.7 Vict Descent

Herkunft des Opfers

- A - Other Asian
- B – Black
- C – Chinese
- D – Cambodian
- F – Filipino
- G – Guamanian
- H - Hispanic/Latin/Mexican
- I - American Indian/Alaskan Native
- J – Japanese
- K – Korean
- L – Laotian
- O – Other
- P - Pacific Islander

- S – Samoan
- U – Hawaiian
- V – Vietnamese
- W – White
- X – Unknown
- Z - Asian Indian

2.1.8 Premis Desc

Beschreibung der Räumlichkeit des Verbrechens (Straße, Laden, Parkplatz etc.)

2.1.9 Weapon Desc

Beschreibung der verwendeten Waffe, falls eine bekannt.

2.2 Falsche Dateninstanzen

Bei dem Datensatz der LAPD handelt es sich um Originaldaten der Kriminalberichte (crime reports), die auch auf Papier aufgenommen worden sind. Dies führt zu einigen Ungenauigkeiten der Daten. Fehlende Einträge werden mit „0°, 0°" markiert. Adressenangaben sind gezielt gefälscht, um private Daten zu schützen. Ein Überblick über den Datenverlauf liefert folgendes Bild.

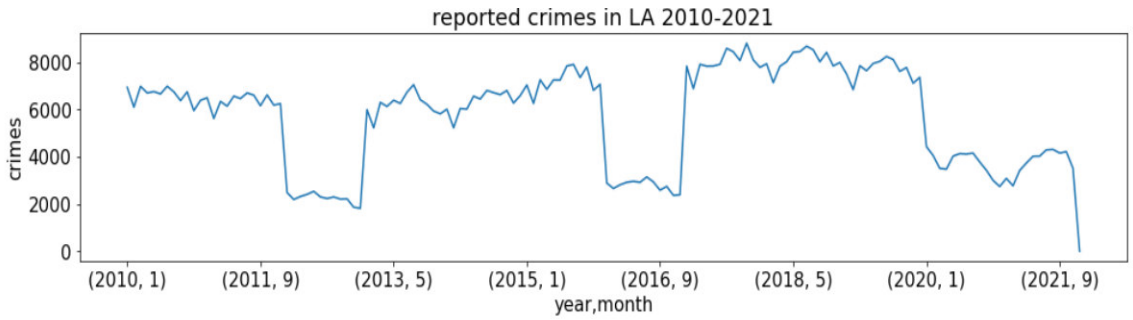


Bild 1: Datenverlauf der reported crimes über dem Zeitraum des Datensatzes

Es sind Unstetigkeiten im Verlauf festzustellen. Dies weist auf fehlende Dateneinträge hin. Eine Datenanalyse mit Hilfe von Klassifizierern sollte jedoch deswegen kein Problem darstellen, da die Datenmenge und deren Zuordnung zu den entsprechenden Attributen ausreichend sei.

3 Data Preprocessing

Als Data Preprocessing wird die Vorbereitung der Daten vor der eigentlichen Datenanalyse bezeichnet. Durch diese Datenaufbereitung können unterschiedliche Datenanpassungen vorgenommen werden.

Provost (Provost, 2015) beschreibt die wichtigen Aufgaben der Datenaufbereitung. Generell handelt es sich um eine Überführung der Daten in Form und Format, was die Analysemethoden verarbeiten können. Ergänzung und Entfernung fehlender oder falscher Daten gehört dazu. Manche Modelle sind für symbolische und kategoriale Daten ungeeignet. Dies bedingt eine Umwandlung in numerischen Werte. „Darüber hinaus müssen numerische Werte oft normalisiert oder skaliert werden, damit sie vergleichbar sind."

3.1 Allgemeine Anpassungen und Entfernung irrelevanter Daten

Der Datensatz verfügt über ausreichend viele Einträge (über 2 Millionen). Dies gibt die Möglichkeit ungeeignete Dateninstanzen „großzügig" zu entfernen.

3.1.1 Fehlende Opferdefinition

Essentiell für die nachfolgende Datenanalyse ist die vollständige Definition des Opfers. Somit können alle Einträge ohne Opfer oder mit fehlerhaften Eingaben gelöscht werden. D.h. bei fehlendem oder fehlerhaftem Opferalter, -Geschlecht oder –Herkunft wird die Zeile gelöscht.

Diese Datenreduktion führt nahezu zu einer Halbierung der Datenmengen.

3.1.2 Gruppierung in Altersgruppen

Ein weiteres auf das Opfer bezogenes Attribut ist die Spalte Victim_Age. Das konkrete Opferalter stellt an sich eine große Varianzerhöhung und Datenverzerrung dar. Für eine Klassifikation würde eine Unterteilung in Altersgruppen mehr Sinn machen. Es werden folgende Gruppen festgehalten:

- <18
- 19-30
- 31-40
- 41-50
- 51-60
- >60

3.1.3 Gruppierung der Herkünfte in Hauptgruppen

Die Spalte Vict_Descent stellt eine weitere unnötige Verzerrung des Datensatzes dar. In 2.1.7 ist dargestellt dass es sich um 19 unterschiedliche Möglichkeiten handelt. Diese werden in folgende 5 Hauptgruppen aufgeteilt:

- White → 1
- Black → 2
- Latin → 3
- Asian → 4
- Others → 5

3.1.4 Opfergeschlecht als numerisches Wert

Um die volle Funktionsfähigkeit der verwendeten Analysemethoden zu gewährleisten sollen alle Geschlechter in nummerischen Werten übertragen werden.

- M → 1 (männlich)
- F → 2 (weiblich)
- X → 0 (andere)

In den neu entstandenen Geschlechtsgruppen finden die Daten folgende Verteilung (s. Bild 2).

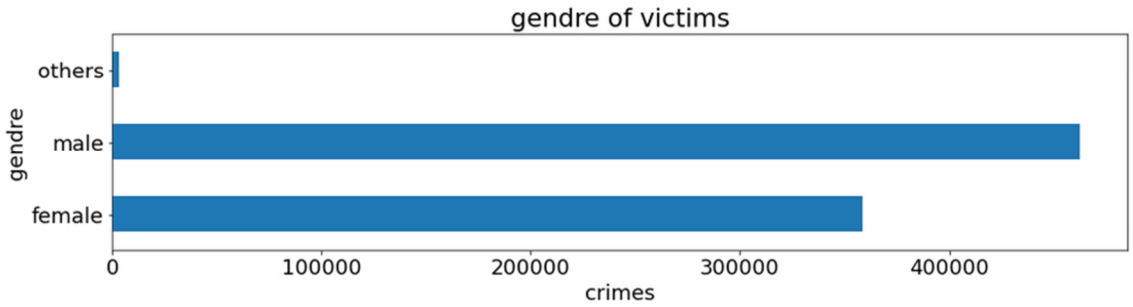


Bild 2: Datenverteilung nach Geschlecht

Das auf Bild 3 dargestellte Histogramm zeigt die Verteilung der Verbrechenshäufigkeit über das Opferalter geteilt nach Opfergeschlecht.

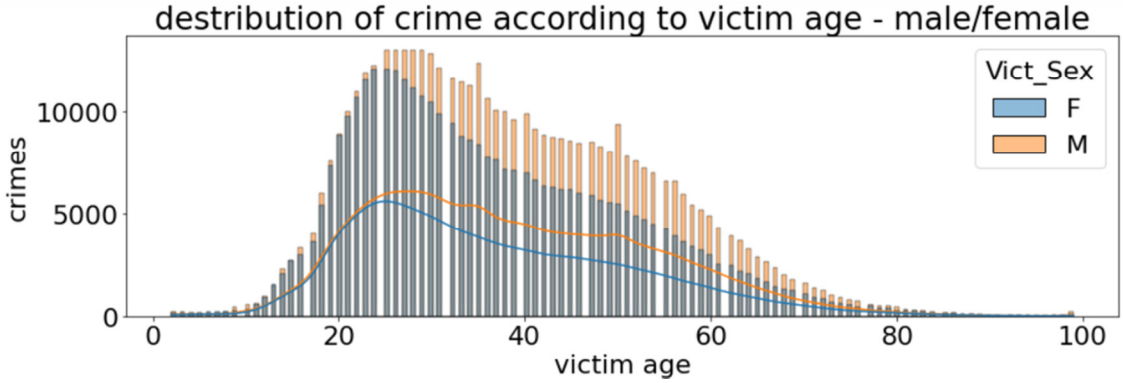


Bild 3: Verteilung der Verbrechen über Opferalter mit Unterscheidung nach Opfergeschlecht

3.1.5 Entfernung irrelevanter Locations

Die Aufgabenstellung beschreibt eine nur für Stadtbesucher und Touristen angepasste Datenanalyse. Dies gibt die Möglichkeit weitere, für diese Gruppen irrelevante Locations zu entfernen. Einige repräsentative Beispiele sind wie folgt:

- ABORTION CLINIC/ABORTION FACILITY*
- CHEMICAL STORAGE/MANUFACTURING PLANT
- COLLEGE/JUNIOR COLLEGE/UNIVERSITY
- DAM/RESERVOIR
- DAY CARE/CHILDREN*
- DEPT OF DEFENSE FACILITY
- DETENTION/JAIL FACILITY
- ELEMENTARY SCHOOL
- HOSPITAL
- MANUFACTURING COMPANY
- METHADONE CLINIC
- MORTUARY
- OFFICE BUILDING/OFFICE
- POLICE FACILITY
- HIGH SCHOOL
- Usw.

Soll die Räumlichkeit in dieser Liste zu finden sein, wird die komplette Instanz entfernt.

3.1.6 Definition von Tageszeiten aus der Uhrzeit des Verbrechens

Die genaue Uhrzeit, wann das Verbrechen geschehen ist, ist für die Datenauswertung nicht von großer Bedeutung. Eine Gruppierung in Tageszeiten würde die Varianz reduzieren. Die Hauptgruppen Tag und Nacht werden eingeführt.

- Day → 0 (06:00 – 18:00)
- Night → 1 (18:00 -06:00)

In Bild 4 ist die Verteilung der Daten nach Tageszeiten dargestellt.

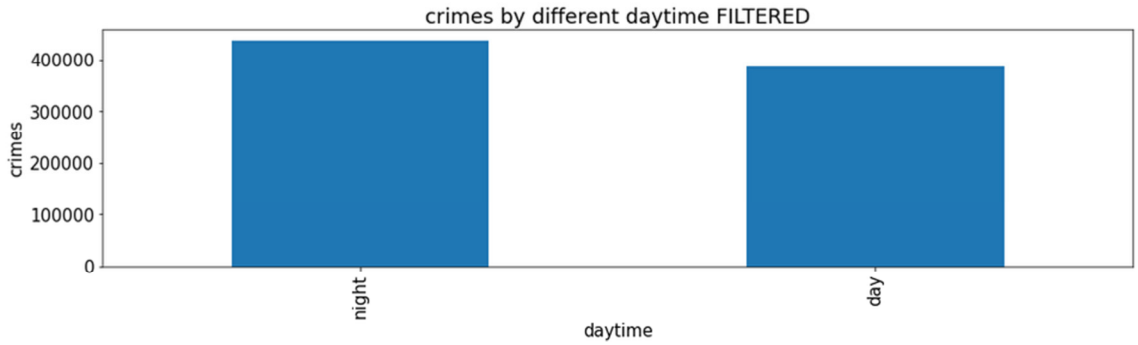


Bild 4: Verteilung der Daten nach Tageszeit Tag/Nacht

3.1.7 Gruppierung der LA-Areas zu Communities

Die LAPD unterscheidet zwischen 21 Gebieten in der Stadt. Diese lassen sich mit Hilfe folgender Karte in 4 Communities zusammenfassen.

- valley → 0
- west → 1
- central →2
- south → 3

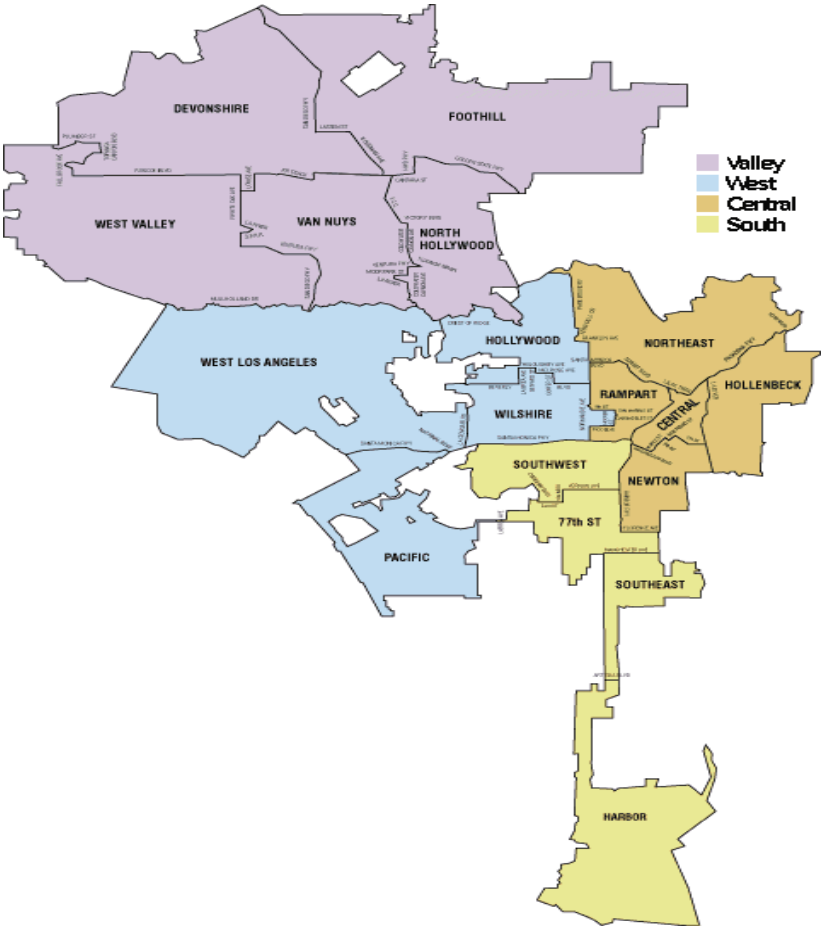


Bild 5: Karte der 21 Polizeigebiete in 4 Communities (www.lapdonline.org, 2022)

Auch bei diesen Gruppen könnte es zu einer Verfälschung kommen. Interessant wären noch Informationen wie Industriegebiet, Wohngebiet, Partygebiet, etc. In Bild 6 ist die Datenverteilung nach der Gruppierung gezeigt.

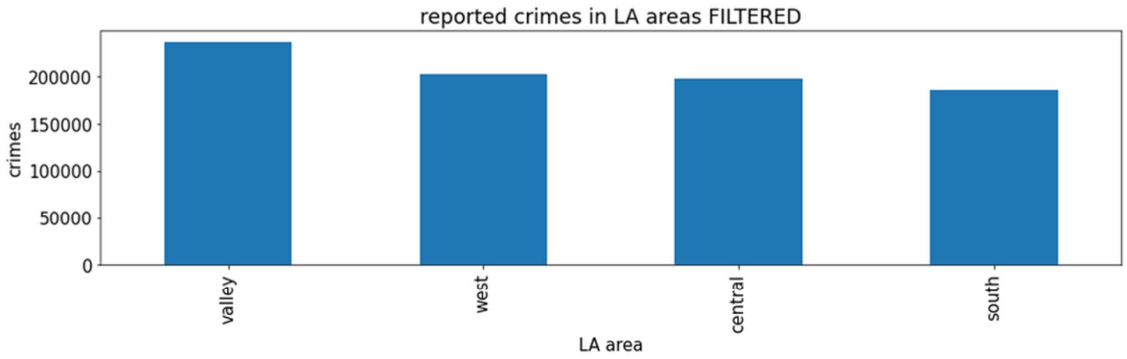


Bild 6: Datenverteilung Areas nach der Gruppierung

3.1.8 Zusammenfassung zu Public Transportation

Die Verbrechen in den öffentlichen Verkehrsmitteln sind für die vorliegende Arbeit viel zu ausführlich dokumentiert. Es werden Metro-Linien, Bus-Linien oder Haltestellen unterschieden. Um diese unnötige Varianz zu reduzieren werden alle diese Instanzen in der Hauptgruppe PUBLIC_TRANSPORTATION zusammengefasst.

3.1.9 Gruppierung der Räumlichkeiten in Hauptgruppen

Das Attribut Premis_Desc verfügt über 140 Definitionsmöglichkeiten für die Räumlichkeit des geschehenen Verbrechens. Für diese Gruppierung soll jedoch ein gewisses Fachwissen eingesetzt werden, denn eine falsche Sortierung kann zu Datenverfälschung führen. Die Instanzen werden in 5 Hauptgruppen zusammengefasst.

- outdoors
- vehicle_related
- publ_transportation
- store
- others

Für die manuelle Gruppierung werden nur die Elemente betrachtet, die mehr als 1000 Einträge im Datensatz haben. Alle anderen werden der Gruppe „others“ zugeordnet.

Die Gruppierung ergibt folgende Datenstruktur (s. Bild 7).

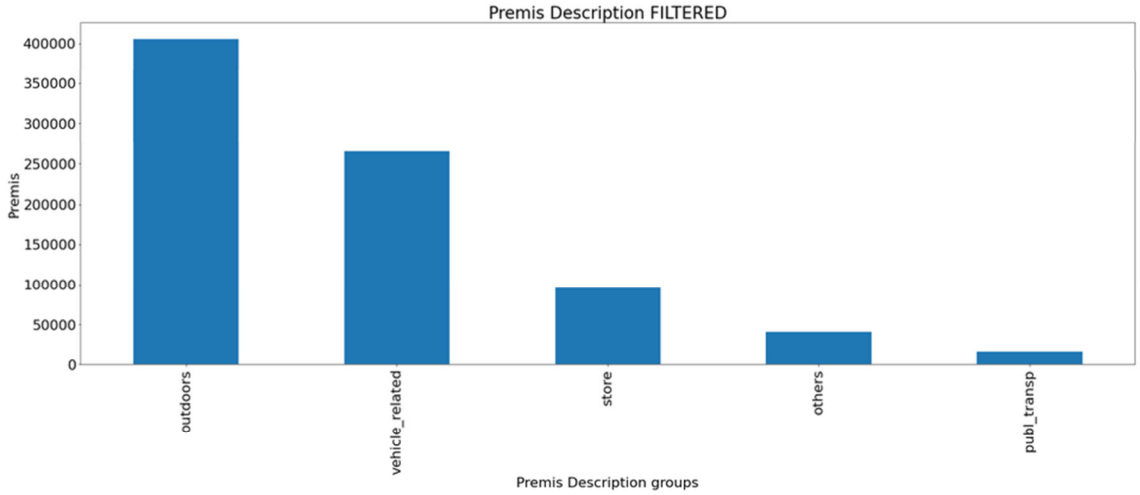


Bild 7: Verteilung der Räumlichkeiten nach Gruppierung

3.1.10 Gruppierung der Verbrechen in Hauptgruppen

Das Attribut Crm_Cd_Desc steht für Crime Code Description und stellt eine standardisierte Beschreibung des Verbrechens dar. Nach der Vorgabe der U. S. Department of Justice werden 104 unterschiedliche Verbrechensarten unterschieden. Diese sollen folgend in 4 Hauptgruppen aufgeteilt werden.

- theft
- assault
- vandalism
- others

Ähnlich wie beim Feature Premis_Desc ist eine solche Sortierung nur mit Vorsicht zu genießen. Auch hier werden nur Elemente mit mehr als 1000 Einträgen berücksichtigt.

Nach der Umgruppierung ergibt sich folgende Datenverteilung (s. Bild 8).

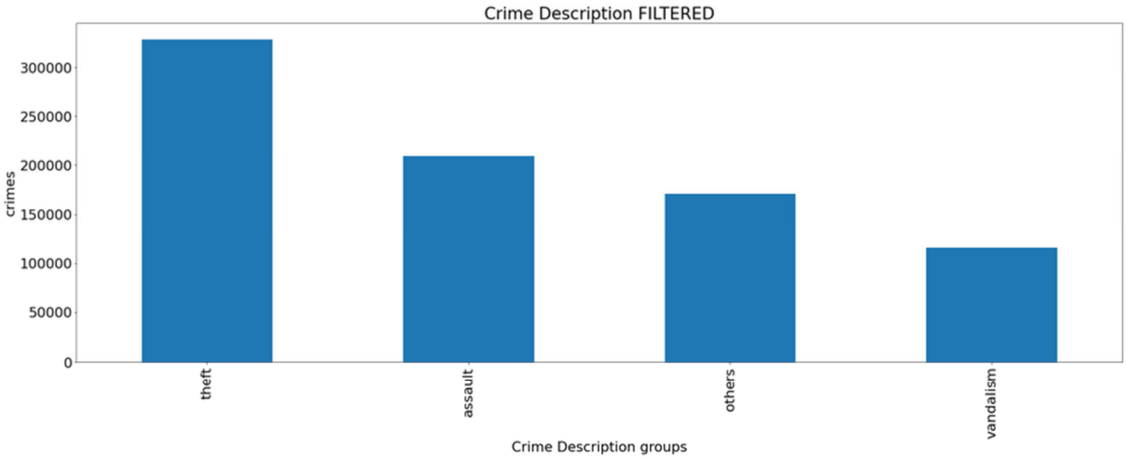


Bild 8: Verteilung der Crimes nach Gruppierung

3.1.11 Verteilung mit und ohne Waffe

Die Gruppierung der Daten nach Waffe erfolgt nach dem einfachen Prinzip, dass wenn eine Waffe vorhanden ist, dann handelt es sich um ein bewaffnetes Verbrechen. Sonst gilt der Fall als unbewaffnet.

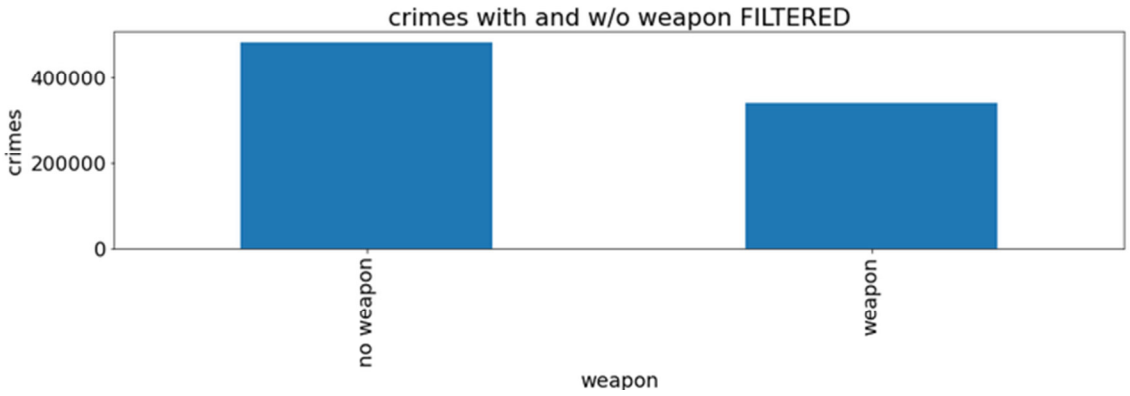


Bild 9: Verteilung der Daten nach Waffe

4 Datenanalyse

Das Ziel der vorliegenden Arbeit ist, ein potentielles Opfer von Kriminalität in Los Angeles vor Gefahr zu warnen. Das Opfer charakterisiert sich nur mit den Features Alter, Geschlecht und Herkunft. Das Datenanalysemodell soll dann für diese drei Eingangswerte das gefährlichste Stadtgebiet, die wahrscheinlichste Tageszeit, die Art des Überfalls, die Räumlichkeit und ob der Übergriff bewaffnet oder nicht sein wird, ermitteln.

Bild 10: Graphische Darstellung der Klassifikationsaufgabe

Dafür werden folgende Klassifizierungsmethoden eingesetzt:

- K-Neighbors Classifier
- Decision Tree Classifier
- Random Forest Classifier
- SVC (Support Vector Classifier)
- Ada Boost Classifier
- Voting Classifier (Ensemble Methode).

Die Daten (ca. 800.000 Instanzen) werden in einem Verhältnis von 25/75 in Trainings- und Testdaten aufgeteilt. Im ersten Schritt erfolgt das Modelltraining mit den Default-Werten in Python. Folgende Einschränkungen sind aufgrund erhöhter Berechnungszeit einzuführen:

- Zum Training und Testen von SVC werden nur 30% des Datensatzes verwendet.
- SVC wird von der Ensemble Methode ausgeschlossen.

Die Genauigkeit jeder Methode wird mit Hilfe der Akkuranz durch den Vergleich der vorhergesagten Werte und der Testwerten bewertet. Diese lässt sich wie folgt berechnen:

$$Akkuranz = \frac{\text{Anzahl korrekter Klassifizierungen}}{Anzahl\ korrekter\ Klassifizierungen + Anzahl\ falscher\ Klassifizierungen}$$

Eine Übersicht der Ergebnisse liefert das Diagramm in Bild 11.

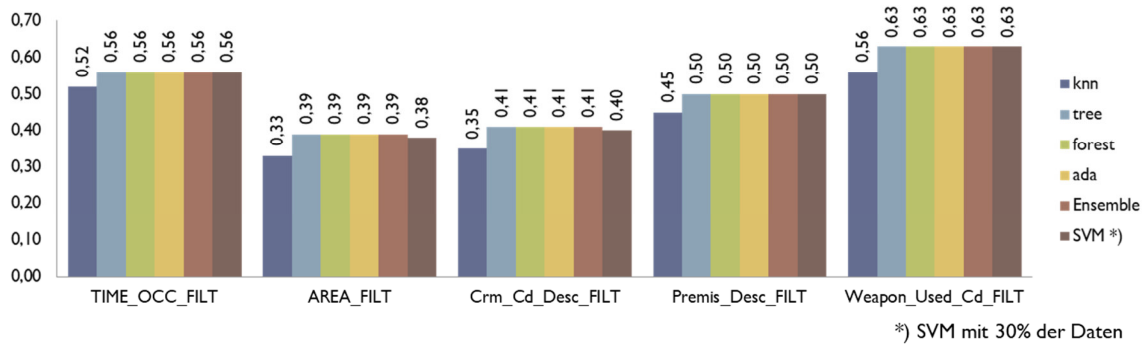


Bild 11: Akkuranzen der Datenanalysemethoden

Es ist ersichtlich, dass alle Methoden eine geringe Treffgenauigkeit ausweisen. Weiterhin ist die Akkuranz bei Attributen mit zwei Elementen (Tageszeit und Waffe) am höchsten. Dies lässt sich durch die extrem reduzierte Varianz erklären.

Die Features, bei denen eine „logische“ Aufteilung in Hauptgruppen durchgeführt ist, liegen niedriger in ihrer Treffgenauigkeit. Die Attribute Area, Crime Description und Premis Description wurden nach Kriterien sortiert, ohne dabei über das benötigte Fachwissen zu verfügen. Eine mögliche Erklärung der schlechteren Ergebnisse dabei ist die falsche Zuordnung in neuen Gruppen. Weiterhin ist zu vermuten, dass die Abhängigkeit zwischen Input und Output nicht vorhanden ist. Man spricht in diesem Fall von prädiktionsschwache X-Werten.

Es fällt auf, dass das Attribut Area besonders schlecht abschneidet. Bei der Gruppierung sind diverse Informationen nicht vorhanden (Wohngebiet, Industriezone, Immobilienpreise, Party-Locations etc.). Somit ist auch hierbei eine fälschliche Definition der Hauptgruppen zu vermuten.

Im Allgemeinen gilt auch, dass der Informationsgehalt des Datensatzes bezüglich dem Opfer von Kriminalität sehr gering ist. Es sind nur drei Faktoren für die Definition einer Targetgruppe vorhanden, und diese sind nicht ausreichend.

Des weiteren soll eine Hyperparameteroptimierung durchgeführt werden. Dadurch werden Einstellparameter der Analysemethoden optimiert, um deren Treffgenauigkeit zu verbessern.

4.1 Hyperparameteroptimierung

Die Hyperparameteroptimierung stellt ein rechenintensives Verfahren dar. Die Parameterwerte erzeugen durch ihre Kombinationen eine mehrdimensionale Matrix, in der jedes Element eine Optimierungsrechnung darstellt. Aufgrund der Berechnungsdauer wird die Optimierung nur anhand der Methoden K-Neighbors Classifier und Decision Tree Classifier durchgeführt. Des weiteren wird für diese Berechnungen die Datenbasis reduziert und nur 30% des Datensatzes verwendet. Als Optimierungsverfahren wird Random Grid Search eingesetzt (GridSearchCV). Folgende Parameter sollen variiert werden:

- K-Neighbors Classifier: weights = ['uniform', 'distance‘]

  n_neighbors = list(range(15, 31)

- Decision Tree Classifier: criterion = ['gini', 'entropy']

  max_depth = [2, 4, 6, 8, 10, 12]

Die Ergebnisse der Hyperparameteroptimierung sind in Bild 12 dargestellt. Vergleichsmethode ist wiederum die Akkuranz.

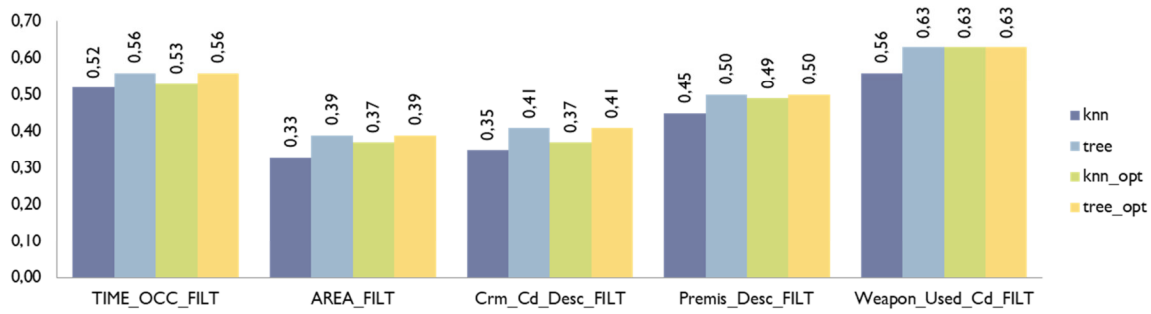


Bild 12: Ergebnisse der Hyperparameteroptimierung für die Methoden K-Neighbors Classifier und Decision Tree Classifier

Ein Vergleich der Genauigkeit von Decision Tree zwischen Bild 11 und Bild 12 zeigt leichte Unterschiede. Diese lassen sich durch die reduzierte Datenmenge bei der Optimierung erklären. Das Diagramm auf Bild 12 soll somit für sich alleine interpretiert werden. Es ist ersichtlich, dass eine Verbesserung der Performance erreicht wurde. Jedoch kann kein deutlicher Sprung in der Methodengenauigkeit erzielt werden.

Da dieser „letzte Schliff" der Analyse keine signifikante Verbesserung mit sich bringt, wird aufgrund der erhöhten Berechnungsdauer darauf verzichtet. Da sich Decision Tree als eine bezüglich der Ergebnisse zufriedenstellende Methode einsortieren lässt und zugleich eine akzeptable Berechnungszeit hat, soll diese auch weiterhin in der vorliegenden Arbeit verwendet werden.

5 Interpretation

Um die Prädiktionsgenauigkeit zu interpretieren wird die Wahrheitsmatrix (Confusion Matrix) eingeführt. Dies ist eine Darstellung der vorhergesagten Werte im Vergleich zu den Tatsächlichen. Somit kann für jede Klasse abgeleitet werden wieviel Elemente richtig oder falsch vom Klassifikator zugeordnet sind. Sie ist also in der Lage die Qualität des Klassifikator-Outputs zu bewerten. Eine exemplarische Matrixstruktur für zwei Klassifikationsgruppen ist in Bild 13 zu sehen [Koch, 2022]

		Tatsächliche Klasse	
		Positiv	Negativ
Vorhergesagte Klasse	Positiv	True-Positive (TP)	False-Positive (FP)
	Negativ	False-Negative (FN)	True-Negative (TN)

Bild 13: Exemplarische Darstellung der Wahrheitsmatrix für zwei Klassen (Koch, 2022)

Es sind 4 Quadrante zu erkennen:

- True Positive: Vorhersage für die Gruppe „Positiv“ ist richtig
- False Positive: Vorhersage für die Gruppe „Positiv“ ist falsch
- False Negative: Vorhersage für die Gruppe „Negativ“ ist falsch
- True Negative: Vorhersage für die Gruppe „Negativ“ ist richtig.

Bei mehreren Klassen wie z.B. LA-Areas im betrachteten Beispiel wachsen die Matrixdimensionen entsprechend. Im Folgenden soll die Confusion Matrix für die betrachtete Analysemethode Descision Tree für alle Ziel-Attribute ermittelt werden.

5.1 Prädiktion der Tageszeiten

Die Confusion Matrix für das Feature Tageszeit ist in Bild 14 zu sehen. Die berechneten Werte sind relativ.

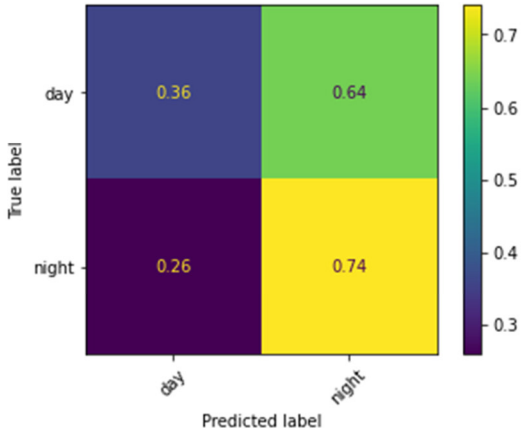


Bild 14: Confusion Matrix für das Attribut Tageszeit für Descision Tree

Der Match Tag-Tag oder Nacht-Nacht in den Matrixdimensionen True und Predicted stellt die richtig vorhergesagten Tageszeiten. Das Modell berechnet also mit 74% ein Verbrechen in der Nacht zuversichtlicher (74% der Testdaten sind richtig zugeordnet). Bei der Tagesklasse sind es nur 36%.

5.2 Prädiktion der LA-Areas

Vier LA-Areas sind entsprechend in einer 4x4-Matrix darstellbar (s.Bild 15).

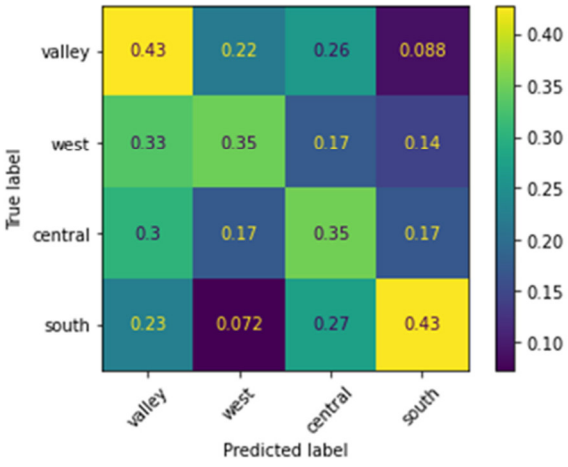


Bild 15: Confusion Matrix für das Attribut LA-Area für Descision Tree

Es ist ersichtlich, dass die relativen Werte in der Diagonale oben-links nach unten-rechts die Größten in der Darstellung sind. Dies bedeutet, dass die Klassifizierung in der richtigen Klasse am wahrscheinlichsten ist. Nichtsdestotrotz liegen die Werte mit 35% bis 43% sehr niedrig und die Mehrheit aller Prädiktionen ist falsch.

5.3 Prädiktion der Art des Überfalls

Das Feature „Crime Code Description“ wurde in vier Hauptgruppen aufgeteilt, was auch eine 4x4-Matrix ergibt (s. Bild 16).

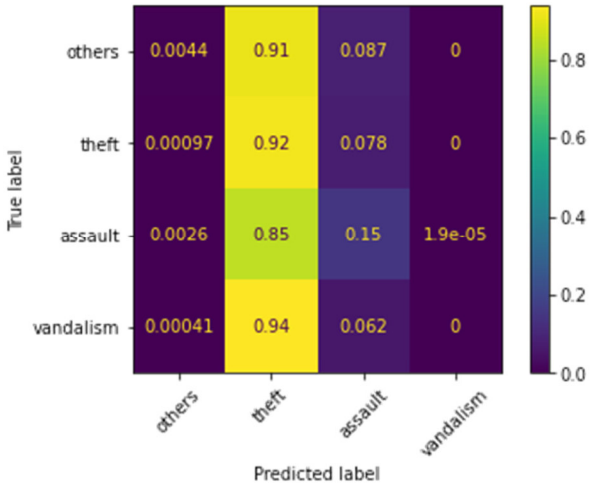


Bild 16: Confusion Matrix für das Attribut Crime Code Description für Descision Tree

Offensichtlich ist der Performanceunterschied zwischen der Gruppe Raubüberfall (theft) und allen Anderen. Die Analysemethode tendiert dazu alle Fälle dieser Klasse zuzuordnen, die auch die größte Gruppe darstellt. Dies kann ein Zeichen für prädiktionsschwache X-Werte sein. Da das Modell keine ausreichende Abhängigkeit zwischen X und Y findet, erfolgt die Zuordnung zu der allergrößten Klasse. Da es sich um die Gruppe mit der größten Datendichte handelt ist es auch am wahrscheinlichsten in dieser Gruppe eine richtige Prädiktion zu erreichen. Dies bestätigt sich auch durch die komplett fehlende Treffsicherheit für das Attribut Vandalismus, das, wie auf Bild 8 gezeigt, die kleinste Gruppe darstellt. Nähere Analysen dazu sind in Kapitel 5.6 zu finden.

Dazu kommt noch die Fehlerquelle der falschen Zuordnung zu den vier Hauptgruppen.

5.4 Prädiktion der Räumlichkeit des Überfalls

Das Attribut „Premis Description“ hat eine sehr ähnliche Struktur wie „Crime Code Description“. Mit 5 Klassen ergibt es auch die in Bild 17 dargestellt 5x5-Matrix.

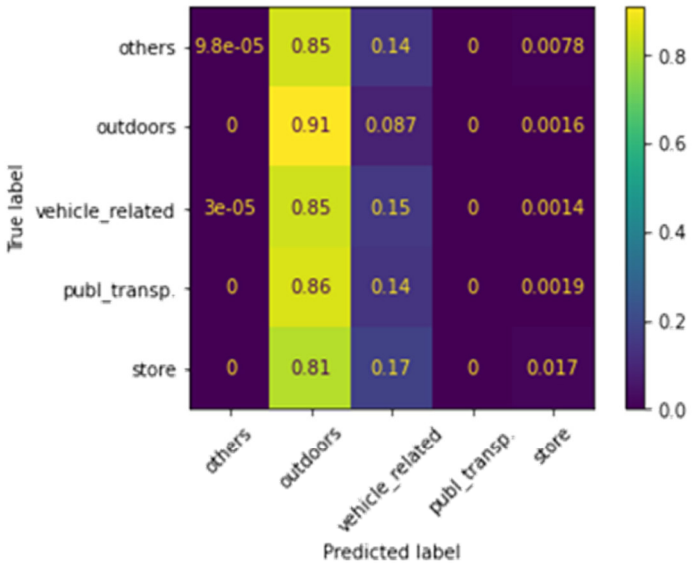


Bild 17: Confusion Matrix für das Attribut Premis Description für Descision Tree

Es zeichnet sich ein ähnliches Verhalten wie bei der Verbrechensbeschreibung. Der größten Gruppe „outdoor“ kann eine sehr hohe Treffsicherheit von 91% zugeordnet werden aber das Modell klassiert fast alle Inputs zu dieser Gruppe. Dagegen ist die kleinste Gruppe „Publik Transportation“ leer. Wegen der Prädiktionsschwäche der X-Werte lässt sich keine modellbasierte Datenzuordnung feststellen (Näheres – s. Kapitel 5.6).

5.5 Prädiktion bewaffnet oder nicht

Der letzte Output des Modells ist das Attribut „Weapon Descroption“, was zum zweidimensionalen Feature bewaffnet oder nicht bewaffnet reduziert wurde. Die Confusion Matrix sieht wie folgt aus.

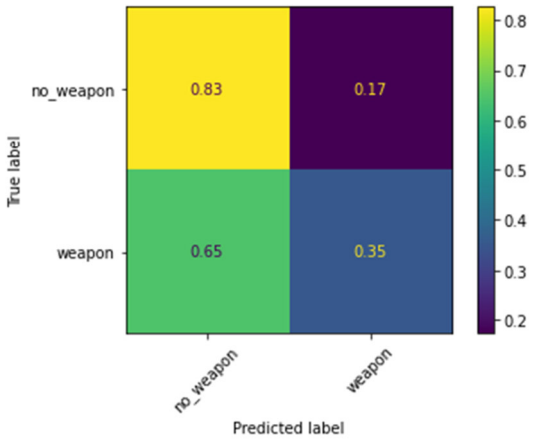


Bild 18: Confusion Matrix für das Attribut Weapon Description für Descision Tree

Die Grafik zeigt eine gute Treffsicherheit für die Klasse „ohne Waffe“ von 83%, die auch die Hauptgruppe in der Datenverteilung darstellt.

5.6 Blackboxinterpretation mit Hilfe von SHAP

Das Framework SHAP (SHapely Additive exPlanations) ist eine Bewertungsmethode, um den Einfluss der Input-Attribute zu quantifizieren. Die Methode stellt die Veränderung des

Ausgangsfeature bei einer Variation der Eingänge (Koch, 2022). Die Diagramme sind in Bild 19 untereinander in einer Graphik dargestellt.

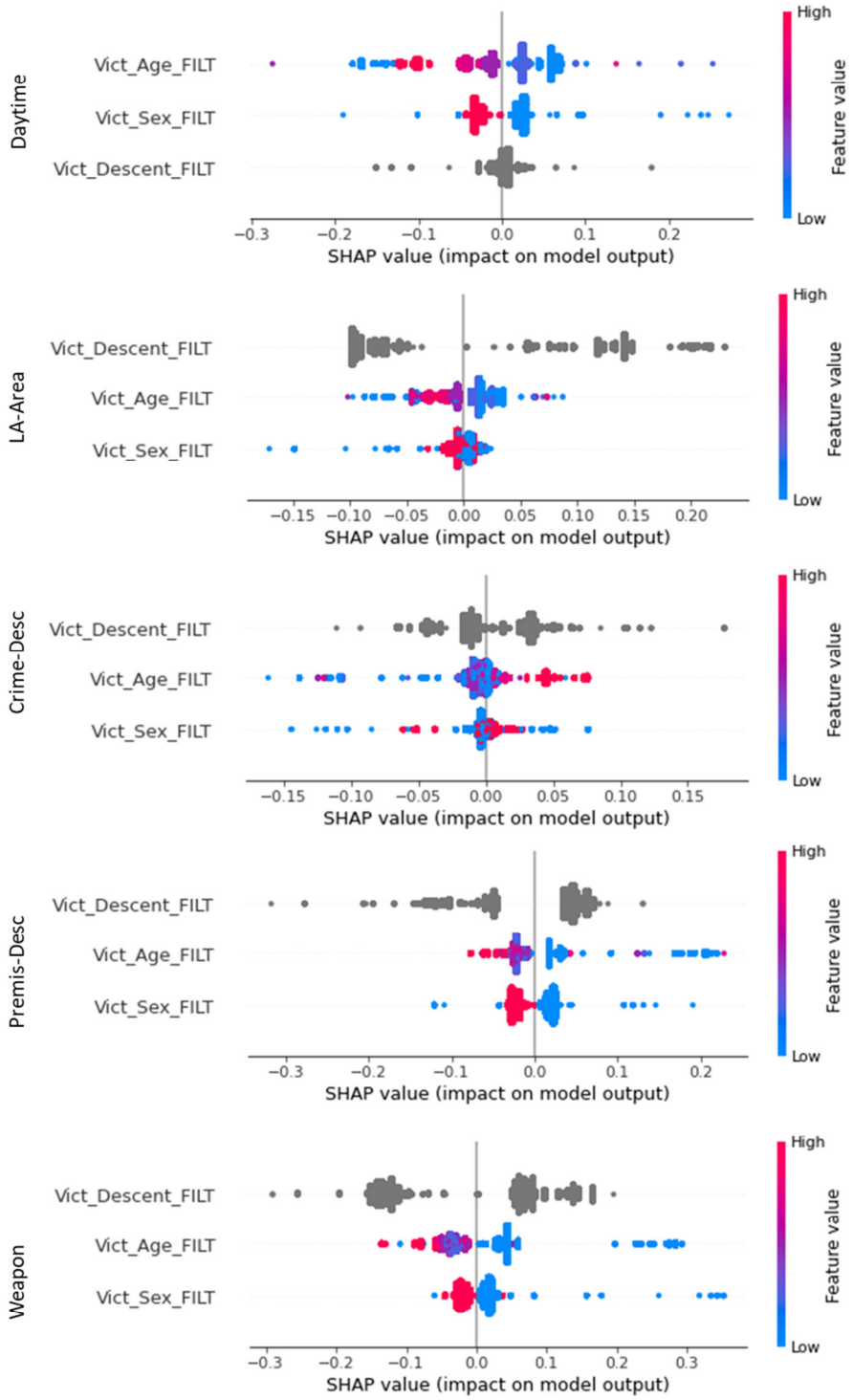


Bild 19: SHAP-Plots für Descision Tree

Unter Berücksichtigung der unterschiedlichen Skallierung der X-Achse ist es ersichtlich, dass bei den Features Crime-Desc und Premis-Desc eine intensivere Anhäufung von Punkten um die Null-Achse zu beobachten ist. Dies erklärt die auffällige Prädiktionsschwäche der X-Werte für diese Attribute.

6 Zusammenfassung und Ausblick

Die vorliegende Arbeit beschäftigt sich mit der Datenanalyse der Verbrechensberichte in der Stadt Los Angeles. Das Ziel besteht darin, eine Vorhersage für eine Person zu erstellen, um wahrscheinliche Kriminalitätsparameter als Gefahren zu ermitteln. Als Zielgruppe der Datenanalyse werden Touristen und Besucher der Stadt gesetzt. Dies soll in der Programmierumgebung von Python geschehen.

Die Datenaufbereitung besteht grundsätzlich aus drei Schritten. Zuerst soll die Datenmenge reduziert werden. Dazu werden vorerst alle Instanzen ohne ein eindeutig definiertes Opfer gelöscht. Weiterhin werden Locations und Verbrechensarten, die für Stadtbesucher irrelevant sind, aussortiert. Die dritte Aufgabe besteht darin die Vielzahl an Elementen in bestimmten Attributen in logischen Hauptgruppen einzuordnen, um die Datenvarianz zu reduzieren. Um die Datenverarbeitung durch die Analysemethoden zu ermöglichen sollen dazu kategorielle Features in numerische Werte übertragen werden.

Das Ergebnis der Datenaufbereitung soll graphisch veranschaulicht werden. Dazu werden Diagramme mit spezifischen Datengruppierungen erstellt. Die Visualisierung der Daten stellt eine erste Überprüfung der Sinnhaftigkeit und der Richtigkeit des Data Preprocessings dar.

Die eigentliche Aufgabe besteht darin Prädiktionsmodelle zu trainieren und diese in deren Genauigkeit zu bewerten und optimieren. Dafür werden die Klassifikatoren K-Neighbors Classifier, Decision Tree Classifier, Random Forest Classifier, SVC (Support Vector Classifier), Ada Boost Classifier, Voting Classifier (Ensemble Methode) eingesetzt. Deren Performance wird mit Hilfe der Akkuranz bewertet.

Die erreichte Akkuranz liegt maximal im Bereich von 0.6. Dies bedeutet, dass das Modell ca. 60% aller Prädiktionen treffsicher als richtig klassiert, was relativ niedrig liegt. Die Hyperparameteroptimierung liefert minimale Verbesserung von einigen Hundertsteln und kann das Hauptproblem nicht lösen. Dafür können einige Gründe definiert werden. Erstens fehlen wichtige Informationen zu den Features. Am Beispiel der LA-Areas wären diese z.B. Wohngebiet, Industriezone, Immobilienpreise, Party-Locations etc. Ein weiterer Punkt ist die nicht „fachliche“ Vorgruppierung der Elemente der Attribute in Hauptklassen. Eine falsche Zuordnung würde zu Verfälschung führen. Als letztes ist der zu geringe Informationsgehalt des Datensatzes zu betrachten. Die drei Eingangswerte Alter, Geschlecht und Herkunft des Opfers sind nicht ausreichend um eine Targetgruppe eindeutig zu definieren und diese mit hoher Treffsicherheit in den Zielklassen zu platzieren. Zudem führt die Prädiktionsschwäche der ausgewählten X-Werte zu keiner exakten Zuordnung in den definierten Klassen.

Weitere Features, die das Opfer beschreiben, sollen importiert werden, um die Targetgruppe exakter definieren zu können. Des weiteren sollte die Vorgruppierung von Attributen mit dem erforderlichen Fachwissen erfolgen. Ohne diese Optimierungen der Datenbasis sind die Ergebnisse einer Datenanalyse nur mit Vorsicht zu genießen.

7 Literaturverzeichnis

www.latimes.com. (2019).

data.lacity.org. (2022).

www.lapdonline.org. (2022).

Koch, P. (2022). *Kursunterlagen Data Science.*

Provost. (2015). *Data Science für Unternehmen.*

Wikipedia. (2022). *Wikipedia.com*.

8 Abbildungsverzeichnis

9 Anhang

9.1 Anhang A: Python Imports

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import BaggingClassifier,
RandomForestClassifier, VotingClassifier, AdaBoostClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC

from sklearn.model_selection import cross_val_score,
train_test_split, GridSearchCV
```

```
from sklearn import metrics
from sklearn.metrics import accuracy_score, plot_confusion_matrix,
confusion_matrix

import  shap as shap
```

9.2 Anhang B: Data Preprocessing Code

```
LA_crime =
pd.read_csv(r"C:\Users\anton\Desktop\Digethik\04_Projektarbeit\Crime
_Data_from_2010_to_2019.csv", sep=",", decimal= ".")

LA_crime.columns = LA_crime.columns.str.replace(' ','_')
LA_crime_raw=LA_crime

# Delete all reports without a defined victim "unpersionalized"
LA_crime.isnull().sum()
LA_crime=LA_crime.drop(LA_crime[LA_crime.Vict_Age <1].index)
LA_crime=LA_crime.drop(LA_crime[LA_crime.Vict_Sex == ""].index)
LA_crime=LA_crime.drop(LA_crime[LA_crime.Vict_Sex == "H"].index)
LA_crime=LA_crime.drop(LA_crime[LA_crime.Vict_Sex == "N"].index)
LA_crime=LA_crime.drop(LA_crime[LA_crime.Vict_Descent == "-"].index)
LA_crime=LA_crime.drop(LA_crime[LA_crime.Vict_Descent == ""].index)
LA_crime=LA_crime[LA_crime['Vict_Descent'].notna()]

#Delete all reports for crime on a private location
private_loc=
pd.read_csv(r"C:\Users\anton\Desktop\Digethik\04_Projektarbeit\priva
te_locations.csv", sep=';', header=None)
for row in private_loc[0]:
    LA_crime=LA_crime.drop(LA_crime[LA_crime.Premis_Desc ==
row].index)

# Delete all irrelevant crimes
irrelevant_crimes=
pd.read_csv(r"C:\Users\anton\Desktop\Digethik\04_Projektarbeit\irrel
evant_crimes.csv", sep=';', header=None)
for row in irrelevant_crimes[0]:
    LA_crime=LA_crime.drop(LA_crime[LA_crime.Crm_Cd_Desc ==
row].index)

# Metropolitan Transportation Authority MTA
# unite all crimes in public transportation to one general location
LA_crime['Premis_Desc'] =
LA_crime.Premis_Desc.str.replace(r'(^.*MTA.*$)',
'PUBLIC_TRANSPORTATION')
LA_crime['Premis_Desc'] =
LA_crime.Premis_Desc.str.replace(r'(^.*SUBWAY.*$)',
'PUBLIC_TRANSPORTATION')
LA_crime['Premis_Cd'] = np.where((LA_crime.Premis_Desc ==
'PUBLIC_TRANSPORTATION'), 900, LA_crime.Premis_Cd)

# handle no location and no weapon
# delete records where no locarion is recorded
LA_crime = LA_crime[LA_crime['Premis_Desc'].notna()]
```

```
#grouping locations to significant
outdoors=["STREET", "SIDEWALK", "PARK/PLAYGROUND", "ALLEY"
"OTHER/OUTSIDE", "BEACH"]
vehicle_related=["PARKING LOT", "VEHICLE, PASSENGER/TRUCK",
"GARAGE/CARPORT", "DRIVEWAY", "PARKING UNDERGROUND/BUILDING", "GAS
STATION"]
publ_transportation=["PUBLIC_TRANSPORTATION", "TRANSPORTATION
FACILITY (AIRPORT)", "BUS STOP"]
store=["RESTAURANT/FAST FOOD","MARKET","OTHER STORE","HOTEL","GAS
STATION","DRUG STORE","MOTEL","HEALTH SPA/GYM","NIGHT CLUB (OPEN
EVENINGS ONLY)","MINI-MART","LIQUOR STORE","LAUNDROMAT","CLOTHING
STORE","BAR/COCKTAIL/NIGHTCLUB","SHOPPING MALL (COMMON
AREA)","COFFEE SHOP (STARBUCKS, COFFEE BEAN, PEET'S, ETC.)","CELL
PHONE STORE","AUTO REPAIR SHOP","BEAUTY/BARBER
SHOP","WAREHOUSE","BAR/SPORTS BAR (OPEN DAY & NIGHT)", "DISCOUNT
STORE (99 CENT,DOLLAR,ETC."]

LA_crime['Premis_Desc_FILT']=LA_crime['Premis_Desc']
#outdoors=1
LA_crime['Premis_Desc_FILT'] =
np.where((LA_crime.Premis_Desc_FILT.isin(outdoors)), 1,
LA_crime.Premis_Desc_FILT)
#vehicle_related =2
LA_crime['Premis_Desc_FILT'] =
np.where((LA_crime.Premis_Desc_FILT.isin(vehicle_related)), 2,
LA_crime.Premis_Desc_FILT)
#publ_transportation =3
LA_crime['Premis_Desc_FILT'] =
np.where((LA_crime.Premis_Desc_FILT.isin(publ_transportation)), 3,
LA_crime.Premis_Desc_FILT)
#store =4
LA_crime['Premis_Desc_FILT'] =
np.where((LA_crime.Premis_Desc_FILT.isin(store)), 4,
LA_crime.Premis_Desc_FILT)
#others=0
LA_crime.Premis_Desc_FILT =pd.to_numeric(LA_crime.Premis_Desc_FILT,
errors ='coerce').fillna(0)

# grouping crimes to segnificant groups
#theft=1
theft=["ROBBERY", "THEFT PLAIN - PETTY ($950 & UNDER)", "THEFT FROM
MOTOR VEHICLE - PETTY ($950 & UNDER)", "THEFT FROM MOTOR VEHICLE -
GRAND ($400 AND OVER)", "INTIMATE PARTNER - SIMPLE ASSAULT", "THEFT-
GRAND ($950.01 & OVER)EXCPT,GUNS,FOWL,LIVESTK,PROD", "BURGLARY",
"THEFT, PERSON", "SHOPLIFTING - PETTY THEFT ($950 & UNDER)",
"ATTEMPTED ROBBERY", "BIKE - STOLEN", "THEFT OF IDENTITY", "VEHICLE
- ATTEMPT STOLEN", "TRESPASSING", "BURGLARY FROM VEHICLE,
ATTEMPTED", "SHOPLIFTING-GRAND THEFT ($950.01 & OVER)", "THEFT FROM
MOTOR VEHICLE - ATTEMPT", "PURSE SNATCHING"]
assault=["BATTERY - SIMPLE ASSAULT", "ASSAULT WITH DEADLY WEAPON,
AGGRAVATED ASSAULT", "INTIMATE PARTNER - SIMPLE ASSAULT", "CRIMINAL
THREATS - NO WEAPON DISPLAYED", "BRANDISH WEAPON", "BIKE - STOLEN",
"THEFT OF IDENTITY", "BATTERY WITH SEXUAL CONTACT", "INTIMATE
PARTNER - AGGRAVATED ASSAULT", "RAPE, FORCIBLE", "OTHER ASSAULT",
"INDECENT EXPOSURE", "CRIMINAL HOMICIDE", "THROWING OBJECT AT MOVING
```

```
VEHICLE", "DISTURBING THE PEACE", "KIDNAPPING"]
vandalism=["VANDALISM - FELONY ($400 & OVER, ALL CHURCH
VANDALISMS)", "VANDALISM - MISDEAMEANOR ($399 OR UNDER)"]

LA_crime['Crm_Cd_Desc_FILT']=LA_crime['Crm_Cd_Desc']
#theft=1
LA_crime['Crm_Cd_Desc_FILT'] =
np.where((LA_crime.Crm_Cd_Desc_FILT.isin(theft)), 1,
LA_crime.Crm_Cd_Desc_FILT)
#assault =2
LA_crime['Crm_Cd_Desc_FILT'] =
np.where((LA_crime.Crm_Cd_Desc_FILT.isin(assault)), 2,
LA_crime.Crm_Cd_Desc_FILT)
#vandalism =3
LA_crime['Crm_Cd_Desc_FILT'] =
np.where((LA_crime.Crm_Cd_Desc_FILT.isin(vandalism)), 3,
LA_crime.Crm_Cd_Desc_FILT)
#others=0
LA_crime.Crm_Cd_Desc_FILT =pd.to_numeric(LA_crime.Crm_Cd_Desc_FILT,
errors ='coerce').fillna(0)

# define with or without weapon
LA_crime.Weapon_Desc.fillna("NO_WEAPON", inplace=True)
LA_crime['Weapon_Used_Cd_FILT']=LA_crime['Weapon_Used_Cd']
LA_crime['Weapon_Used_Cd_FILT'] = np.where((LA_crime.Weapon_Desc ==
'NO_WEAPON'), 0, 1)

# Defining the time into daytime classes
LA_crime.TIME_OCC.apply(pd.to_numeric)
LA_crime['TIME_OCC_FILT']=LA_crime['TIME_OCC']
# night = 1
LA_crime['TIME_OCC_FILT'] = np.where((LA_crime.TIME_OCC_FILT < 600)
| (LA_crime.TIME_OCC_FILT >= 1800), 1, LA_crime.TIME_OCC_FILT)
# day = 0
LA_crime['TIME_OCC_FILT'] = np.where((LA_crime.TIME_OCC_FILT >= 600)
& (LA_crime.TIME_OCC_FILT < 1800), 0, LA_crime.TIME_OCC_FILT)

# Grouping age classes
LA_crime['Vict_Age_FILT']=LA_crime['Vict_Age']
# <18
LA_crime['Vict_Age_FILT'] = np.where((LA_crime.Vict_Age_FILT < 18),
0, LA_crime.Vict_Age_FILT)
# 19-30
LA_crime['Vict_Age_FILT'] = np.where((LA_crime.Vict_Age_FILT >= 18)
& (LA_crime.Vict_Age_FILT < 30), 1, LA_crime.Vict_Age_FILT)
# 31-40
LA_crime['Vict_Age_FILT'] = np.where((LA_crime.Vict_Age_FILT >= 30)
& (LA_crime.Vict_Age_FILT < 40), 2, LA_crime.Vict_Age_FILT)
# 41-50
LA_crime['Vict_Age_FILT'] = np.where((LA_crime.Vict_Age_FILT >= 40)
& (LA_crime.Vict_Age_FILT < 50), 3, LA_crime.Vict_Age_FILT)
# 51-60
LA_crime['Vict_Age_FILT'] = np.where((LA_crime.Vict_Age_FILT >= 50)
& (LA_crime.Vict_Age_FILT < 60), 4, LA_crime.Vict_Age_FILT)
# >60
LA_crime['Vict_Age_FILT'] = np.where((LA_crime.Vict_Age_FILT >= 60),
```

```
5, LA_crime.Vict_Age_FILT)

# SEX: Male: 1, Female: 2, Other: 0
# Delete undefined gendres and Vict_Descent
LA_crime["Vict_Sex_FILT"] = LA_crime["Vict_Sex"].replace({'M': 1,
'F': 2, 'X' : 0, 'N' : 0, 'H' : 0})

#A - Other Asian
#B - Black
#C - Chinese
#D - Cambodian
#F - Filipino
#G - Guamanian
#H - Hispanic/Latin/Mexican
#I - American Indian/Alaskan Native
#J - Japanese
#K - Korean
#L - Laotian
#O - Other
#P - Pacific Islander
#S - Samoan
#U - Hawaiian
#V - Vietnamese
#W - White
#X - Unknown
#Z - Asian Indian
asian=["A", "C", "D", "F", "J", "K", "L", "V", "Z"]
other=["G", "O", "P", "S", "X", "I", "U"]

LA_crime['Vict_Descent_FILT']=LA_crime['Vict_Descent']
#white = 1
LA_crime['Vict_Descent_FILT'] = np.where((LA_crime.Vict_Descent ==
"W"), 1, LA_crime.Vict_Descent_FILT)
#black = 2
LA_crime['Vict_Descent_FILT'] = np.where((LA_crime.Vict_Descent ==
"B"), 2, LA_crime.Vict_Descent_FILT)
#latin = 3
LA_crime['Vict_Descent_FILT'] = np.where((LA_crime.Vict_Descent ==
"H"), 3, LA_crime.Vict_Descent_FILT)
#asian=4
LA_crime['Vict_Descent_FILT'] =
np.where((LA_crime.Vict_Descent.isin(asian)), 4,
LA_crime.Vict_Descent_FILT)
#others=5
LA_crime['Vict_Descent_FILT'] =
np.where((LA_crime.Vict_Descent.isin(other)), 0,
LA_crime.Vict_Descent_FILT)

# grouping areas to communities
valley = [9, 10, 15, 16, 17, 19, 21]
west=[6, 7, 8 ,14, 20]
central=[1, 2, 4, 11, 13]
south=[3, 5, 12, 18]

LA_crime['AREA_FILT']=LA_crime['AREA_']
#valley =0
```

```
LA_crime['AREA_FILT'] = np.where((LA_crime.AREA_.isin(valley)), 0,
LA_crime.AREA_FILT)
#west=1
LA_crime['AREA_FILT'] = np.where((LA_crime.AREA_.isin(west)), 1,
LA_crime.AREA_FILT)
#central=2
LA_crime['AREA_FILT'] = np.where((LA_crime.AREA_.isin(central)), 2,
LA_crime.AREA_FILT)
#south =3
LA_crime['AREA_FILT'] = np.where((LA_crime.AREA_.isin(south)), 3,
LA_crime.AREA_FILT)

# Deleting unneeded features
to_pop = [ "DR_NO", "Date_Rptd", "Rpt_Dist_No", "Part_1-2",
"Mocodes", "Status", "Status_Desc", "Crm_Cd_1", "Crm_Cd_2",
"Crm_Cd_3", "Crm_Cd_4", "LOCATION", "Cross_Street", "LAT", "LON"]

for col in to_pop:
    LA_crime.pop(col)

age_dict = {'<18': 0, '19-30': 1, '31-40' : 2, '41-50' : 3, '51-60'
: 4, '>60' : 5}
sex_dict = {'M': 1, 'F': 2, 'X' : 0}
descent_dict = {'white': 1, 'black': 2, 'latin' : 3, 'asian' : 4,
'others' : 0}
time_dict= {'day': 0, 'night': 1}
area_dict = {'valley': 0, 'west': 1, 'central' : 2, 'south' : 3}
crime_dict =  { 'others' : 0, 'theft': 1, 'assault': 2, 'vandalism'
: 3}
premis_dict =  {'outdoors': 1, 'vehicle_related': 2, 'publ_transp.'
: 3, 'store' : 4, 'others' : 0}
weapon_dict =  {'weapon': 1, 'no_weapon': 0}
```

9.3 Anhang C: Visualisierungscode

```
# VISUALISATION GENERAL CRIME REPORT

plt.rcParams.update({'font.size': 18})
LA_crime['DATE_OCC'] = pd.to_datetime(LA_crime['DATE_OCC'])
fig = plt.figure(figsize=(15,25))


ax1 = fig.add_subplot(511)
ax1.set_ylabel('crimes')
ax1.set_xlabel('Years')
ax1.set_title("reported crimes in LA 2010-2021")
#ax1.plot(LA_crime.groupby([LA_crime.DATE_OCC.dt.year,
LA_crime.DATE_OCC.dt.month])['DATE_OCC'].agg('count'))
LA_crime.groupby([LA_crime.DATE_OCC.dt.year.rename('year'),
LA_crime.DATE_OCC.dt.month.rename('month')])['DATE_OCC'].agg('count'
).plot(ax=ax1, sharex=False)

LA_crime["AREA_FILT"]  =  LA_crime["AREA_FILT"].replace({1:  "valley",
2: "west", 3 : "central", 4 : "south"})
ax2 = fig.add_subplot(512)
ax2.set_ylabel('crimes')
```

```
ax2.set_xlabel('areas in LA')
ax2.set_title("reported crimes in LA areas FILTERED")
LA_crime.groupby([LA_crime.AREA_FILT.rename('LA
area')])['AREA_FILT'].agg('count').nlargest(50).plot(kind='bar')

ax3 = fig.add_subplot(513)
LA_crime["Vict_Sex_FILT"]   =   LA_crime["Vict_Sex_FILT"].replace({1:
"male", 2: "female", 0 : "others"})
ax3.set_ylabel('crimes')
ax3.set_xlabel('crimes')
ax3.set_title("gendre of victims")
LA_crime.groupby([LA_crime.Vict_Sex_FILT.rename('gendre')])['Vict_Se
x_FILT'].agg('count').plot(kind='barh')

LA_crime["TIME_OCC_FILT"]   =   LA_crime["TIME_OCC_FILT"].replace({1:
"day", 2: "night"})
ax4 = fig.add_subplot(514)
ax4.set_ylabel('crimes')
ax4.set_xlabel('daytime')
ax4.set_title("crimes by different daytime FILTERED")
LA_crime.groupby([LA_crime.TIME_OCC_FILT.rename('daytime')])['TIME_O
CC_FILT'].agg('count').nlargest(20).plot(kind='bar')

LA_crime["Weapon_Used_Cd_FILT"]                                    =
LA_crime["Weapon_Used_Cd_FILT"].replace({1:    "weapon",    0:    "no
weapon"})
ax4 = fig.add_subplot(515)
ax4.set_ylabel('crimes')
ax4.set_xlabel('')
ax4.set_title("crimes with and w/o weapon FILTERED")
LA_crime.groupby([LA_crime.Weapon_Used_Cd_FILT.rename('weapon')])['W
eapon_Used_Cd_FILT'].agg('count').nlargest(20).plot(kind='bar')

fig.tight_layout(pad=2.0)

fig = plt.figure(figsize=(15,20))
LA_crime=LA_crime.drop(LA_crime[LA_crime.Vict_Sex_FILT <1].index)

ax1 = fig.add_subplot(411)
#ax1.set_ylabel('crimes')
#ax1.set_xlabel('Years')
ax1.set_title("destribution  of  crime  according  to  victim  age  -
male/female")
sns.histplot(data=LA_crime.Vict_Sex_HIST,
x=LA_crime.Vict_Age.rename("victim          age"),          kde=True,
hue=LA_crime.Vict_Sex, palette="tab10")

plt.ylabel('crimes')
```

9.4 Anhang C: Code zu Datenanalyse

```
# prediction with age, sex, race

import warnings
warnings.filterwarnings('ignore')
```

```
#LA_crime_frac=LA_crime.sample(frac = 0.3, random_state=1)
LA_crime_frac=LA_crime

x_columns=LA_crime_frac[["Vict_Age_FILT",           "Vict_Sex_FILT",
"Vict_Descent_FILT"]]

y_columns=LA_crime_frac[["TIME_OCC_FILT",               "AREA_FILT",
"Crm_Cd_Desc_FILT",  "Premis_Desc_FILT", "Weapon_Used_Cd_FILT"]]
y_columns=LA_crime_frac[["Weapon_Used_Cd_FILT"]]

#methods = [('tree',DecisionTreeClassifier())]
methods         =         [('knn',         KNeighborsClassifier()),
('tree',DecisionTreeClassifier()),
          ('forest',RandomForestClassifier()),
('ada',AdaBoostClassifier())]
#methods = [ ('svm',SVC())]

ls_dict=[time_dict, area_dict, crime_dict, premis_dict, weapon_dict]
idx = 0

for (column_name, column_data) in y_columns.iteritems():

    print(column_name)
    x_train,  x_test,  y_train,  y_test  =  train_test_split(x_columns,
column_data, test_size=0.25)
    for method in methods:
        if method[0] == "knn_":
            k_range = list(range(15, 31))
            weight_options = ['uniform', 'distance']
            param_grid          =          dict(n_neighbors=k_range,
weights=weight_options)
            grid   =   GridSearchCV   (method[1],   param_grid,   cv=5,
scoring='accuracy', return_train_score=False)
            model =grid.fit(x_train, y_train)
            print(model.best_params_)
        elif method[0] == "tree_":
            crit = ['gini', 'entropy']
            max_d = [2,4,6,8,10,12]
            param_grid = dict(criterion=crit, max_depth=max_d)
            grid   =   GridSearchCV   (method[1],   param_grid,   cv=5,
scoring='accuracy', return_train_score=False)
            model =grid.fit(x_train, y_train)
            print(model.best_params_)
        elif method[0] == "forest_":
            estimators= [100, 200]
            max_feat = ['auto', 'sqrt', 'log2']
            param_grid        =        dict(n_estimators=estimators,
max_features=max_feat)
            grid   =   GridSearchCV   (method[1],   param_grid,   cv=5,
scoring='accuracy', return_train_score=False)
            model =grid.fit(x_train, y_train)
            print(model.best_params_)
        elif method[0] == "svm_":
            param_grid  =  {'C':  [0.1,  1,  10],  'gamma':  [1,  0.1,
0.01]}
            grid   =   GridSearchCV   (method[1],   param_grid,   cv=5,
```

```
scoring='accuracy', return_train_score=False)
            model =grid.fit(x_train, y_train)
            print(model.best_params_)
        elif method[0] == "ada_":
            estimators= list(range(45, 56))
            rate= [0.9, 1.0, 1.1]
            param_grid = dict(n_estimators=estimators, learning_rate
= rate)
            grid  =  GridSearchCV  (method[1],  param_grid,  cv=5,
scoring='accuracy', return_train_score=False)
            model =grid.fit(x_train, y_train)
            print(model.best_params_)

        model = method[1].fit(x_train, y_train)

        model_predict = np.round(model.predict(x_test))
        model_accuracy = accuracy_score(model_predict, y_test)
        print("Genauigkeit   von   {}:   {:.02}".format(method[0],
model_accuracy))
        labels = list(ls_dict[idx].keys())
        print(labels)
        plot_confusion_matrix(model,         x_test,         y_test,
normalize="true", xticks_rotation=45)
        plt.xticks(                    list(ls_dict[idx].values()),
list(ls_dict[idx].keys()))
        plt.yticks(                    list(ls_dict[idx].values()),
list(ls_dict[idx].keys()))
        plt.show()
    idx=idx+1
    ensemble = VotingClassifier(methods)
    ensemble.fit(x_train, y_train)
    predict = np.round(ensemble.predict(x_test))
    accuracy = accuracy_score(predict,y_test)
    print("Genauigkeit         von         Ensemble         Method:
{:.02}".format(accuracy))
```

9.5 Anhang D: SHAP Plots

```
# Initialize JavaScript visualizations in notebook environment
shap.initjs()
# Define a tree explainer for the built model
explainer = shap.TreeExplainer(model)
# obtain shap values for the first row of the test data
shap_values = explainer.shap_values(x_test.iloc[0])
#shap.force_plot(explainer.expected_value[0],        shap_values[0],
x_test.iloc[0])

# obtain shap values for the test data
shap_values = explainer.shap_values(x_test)
#shap.force_plot(explainer.expected_value[0],        shap_values[0],
x_test)

shap.summary_plot(shap_values[1], x_test)
```

```
#shap.summary_plot(shap_values[1], x_test, plot_type='bar')
```