5. Übungsblatt zur Vorlesung **WS 2018/2019**
Statistische Methoden der Datenanalyse **Prof. W. Rhode**
Abgabe: 22.11.2018 23:59

Zeit	Raum	Abgabe im Moodle; Mails mit Betreff: [SMD1819]
Di. 10-12	CP-03-150	tobias.hoinka@udo.edu, felix.geyer@udo.edu und jan.soedingrekso@udo.edu
Di. 16-18	CP-03-150	simone.mender@udo.edu und alicia.fattorini@udo.edu
Mi. 10-12	CP-03-150	mirco.huennefeld@udo.edu und kevin3.schmidt@udo.edu

Aufgabe 13: *Simulationskette für Neutrinodetektor* **15 P.**

Verwenden Sie für diese Aufgabe die Pythonbibliothek `pandas`. Füllen Sie die Ergebnisse der einzelnen Teilaufgaben, abgesehen von der Letzte, in ein `DataFrame` und speichern Sie dieses am Ende in einer `hdf5`-Datei `NeutrinoMC.hdf5` mit dem Key `Signal`. Die Ergebnisse der letzten Teilaufgabe sollen in ein eigenes `DataFrame` geschrieben und in der selben `hdf5`-Datei unter dem Key `Background` gespeichert werden. Zum Schreiben einer `hdf5`-Datei besitzt das `DataFrame` die Methode `to_hdf()`.

Wichtig: Die erstellte `hdf5`-Datei soll **nicht** mit abgegeben werden! Das fertige Programm muss ohne die bereits existierende `hdf5`-Datei funktionieren.

a) Signal MC

Der Fluss der Neutrinos ist gegeben durch:

$$\Phi = \Phi_0 \cdot \left(\frac{E}{\mathrm{TeV}}\right)^{-\gamma}.$$

Hierbei ist der spektrale Index $\gamma = 2{,}7$, die untere Energiegrenze 1 TeV und die obere Energiegrenze unendlich.

Simulieren Sie mit Hilfe der Transformationsmethode 10^5 Signalereignisse und speichern Sie diese in dem `DataFrame` unter dem Key `Energy`.

b) Akzeptanz Die Wahrscheinlichkeit, ein Ereignis zu detektieren, ist energieabhängig. Dies muss bei der Simulation mit berücksichtigt werden. Die Detektionswahrscheinlichkeit lässt sich durch folgende Gleichung beschreiben:

$$P(E) = (1 - \mathrm{e}^{-E/2})^3.$$

Nutzen Sie das Neumann'sche Rückweisungsverfahren, um die Detektorakzeptanz für die in Teil a) simulierten Signal-Ereignisse zu berücksichtigen. Speichern Sie die Ergebnisse in Form einer *Maske* (`True-False`-Folge) unter dem Key `AcceptanceMask` in dem `DataFrame`. Stellen Sie das Ergebnis von a) und b) in einem Plot dar. Hinweis: Nutzen Sie eine log-log Darstellung.

c) Energiemessung

Ein realistischer Detektor besitzt nur eine endliche Energieauflösung. Zudem wird die Energie nicht direkt gemessen, sondern mit Hilfe energiekorrelierter Observablen rekonstruiert. Eine solche Observable ist bespielsweise die Anzahl der Photomultiplier, die angesprochen haben (im Folgenden als Hits bezeichnet).

Die Anzahl der Hits lässt sich aus einer Normalverteilung mit folgenden Eigenschaften ziehen:

$$\mathcal{N}(10 \cdot E/\mathrm{TeV}, 2 \cdot E/\mathrm{TeV}).$$

Hierbei bezeichnet $\mathcal{N}(10E/\mathrm{TeV}, 2E/\mathrm{TeV})$ die Normalverteilung mit Mittelwert $10E$ und Standardabweichung $2E$ (E in TeV). Wandeln Sie die Zahl der Hits in eine ganze Zahl um, da nur ganze Anzahlen an Hits auftreten können. Achten Sie ebenfalls darauf, dass für die Anzahl der Hits N nur Werte oberhalb von Null in Frage kommen. Ziehen Sie gegebenenfalls eine neue Zufallszahl, falls diese Bedingung verletzt wird. Simulieren Sie die Anzahl der Hits in Abhänigkeit der zuvor simulierten Energie und speichern Sie die Anzahl unter dem Key `NumberOfHits`. Nutzen Sie hierzu die aus der Vorlesung bekannte Polarmethode, um die Normalverteilung zu realisieren.

d) Ortsmessung

Betrachten Sie im Folgenden einen quadratischen Flächendetektor mit der Kantenlänge 10 Längeneinheiten. Das Signal trifft am Punkt (7,3) auf den Detekor. Die Ortsauflösung ist wiederum energieabhängig. Simulieren Sie die Orte zu den zuvor erzeugten Ereignissen, indem Sie sowohl für die x- als auch für die y-Richtung eine Normalverteilung annehmen. Hierbei ist σ energieabhängig und gegeben durch

$$\sigma = \frac{1}{\log_{10}(N+1)},$$

wobei N die zuvor bestimmte Anzahl der Hits ist. Achten Sie hier wiederum darauf, dass die gezogenen Ereignisse innerhalb des Detektors liegen (ggf. neue Zufallszahlen ziehen). Speichern Sie die Koordinaten der Orte unter den Keys `x` und `y`. Stellen Sie die erhaltenen Orte in einem zweidimensionalen Histogramm dar.

e) Untergrund MC

Die Zahl der erwarteten Untergrund-Ereignisse ist groß im Verhältnis zum erwarteten Signal. Erzeugen Sie einen neues `DataFrame` mit den Keys `NumberOfHits`, `x` und `y`.

Simulieren Sie 10^7 Untergrund-Ereignisse mit folgenden Eigenschaften:

- Der Zehner-Logarithmus der Anzahl der Hits folgt einer Normalverteilung mit $\mu = 2$ und $\sigma = 1$.
- Die x- bzw. y-Koordinaten der Ereignisse sind um den Mittelpunkt des Detektors normalverteilt. Hierbei ist $\sigma = 3$.
- Zwischen der x- und der y-Koordinate besteht eine Korrelation von $\rho = 0.5$.

Stellen Sie die Orte der Untergrundereignisse in einem zweidimensionalen und den Logarithmus der Anzahl der Hits in einem eindimensionalen Histogramm dar.

Würfelt man standardnormalverteilte Zufallszahlen $x^\star$ bzw. $y^\star$, so ergeben sich die normalverteilten Zufallszahlen x bzw. y mit beliebigem μ, σ und Korrelationskoeffizientem ρ aus der Transformation:

$$x = \sqrt{1-\rho^2} \cdot \sigma \cdot x^\star + \rho \cdot \sigma \cdot y^\star + \mu$$
$$y = \sigma \cdot y^\star + \mu.$$

Aufgabe 14: *Hauptkomponentenanalyse (PCA)* **5 P.**

a) Erzeugen Sie mit der Funktion `sklearn.datasets.make_blobs` einen Datensatz. Nutzen sie dabei folgende Einstellungen: n_samples=1000, centers=2, n_features=4, random_state=0. Stellen Sie nun zwei beliebige Dimensonen des Datensatzes in einem Scatterplot dar.

b) Beschreiben Sie kurz die Funktionsweise der Hauptkomponentenanalyse. Geben Sie in Worten und in der richtigen Reihenfolge die notwendigen Berechnungen zur Durchführung der Hauptkomponentenanalyse an.

c) Wenden Sie nun die Hauptkomponentenanalyse auf den in a) erzeugten Datensatz an. Nutzen Sie dazu das Paket `sklearn.decomposition.PCA`. Wie lauten die Eigenwerte der Kovarianzmatrix? Wie interpretieren Sie die Eigenwerte?

d) Histrogrammieren Sie nun x' in jeder Dimension und stellen sie x'_1 und x'_2 in einem Scatterplot dar.