**Data Exploration Project – Report**

Dieses Projekt wird im Rahmen der Vorlesung „Data Exploration Projekt“ bei Nils Zerrer im vierten Semester durchgeführt. Ziel ist es einen Machine Learning Algorithmus anhand beliebiger Daten zu trainieren und eine Klassifikation oder Vorhersage zu erstellen.

In diesem Projekt wird mit Hilfe einer linearen Regression eine Vorhersage – basierend auf den gegebenen Daten – getätigt.

# 1. Der Datensatz: World Happiness Report

Der für dieses Projekt ausgewählte Datensatz kann auf Kaggle unter dieser URL gefunden werden: <https://www.kaggle.com/unsdsn/world-happiness>.

Der „World Happiness Report“ beinhaltet Daten, die aus einer weltweiten Umfrage ermittelt worden sind. Obwohl bereits Daten seit dem Jahr 2012 erhoben werden, werden in diesem Projekt nur die Daten aus dem Jahr 2019 betrachtet.

## 1.1 Größe des Datensatzes

Der Datensatz besteht 156 Zeilen, bei dem jede Zeile einem anderen Land entspricht. Jede Zeile beinhaltet die Daten für die neun Feature: “*Country or region*”, “*Overall rank*”, “*Score*”,” *GDP per capita*”, “*Social support*”, “*Healthy life expectancy*”, “*Freedom to make life choices*”, “*Generosity*” und “*Perceptions of corruption*”.

### 1.1.1 Country or region

Dieses Feature beinhaltet das Land – bzw. die Region – aus dem die Daten erhoben worden sind.

### 1.1.2 **Overall rank**

Dieses Feature gibt die Platzierung des entsprechenden Landes in der Rangfolge für das Jahr 2019 an.

### 1.1.3 Score

In diesem Feature ist die Punktzahl angegeben, die von dem entsprechenden Land in der Bewertung erzielt worden ist. Zehn Punkte hätten hier erreicht werden können.

### 1.1.4 GDP per capita

Dieses Feature gib das Bruttoinlandsprodukt (BIP) pro Person des entsprechenden Landes an. Dieser Wert gibt, wie sehr sich das BIP auf die Bewertung des entsprechenden Landes auswirkt.

### 1.1.5 Soical Support

In diesem Feature befindet sich der Wert zur allgemeinen sozialen Unterstützung im entsprechenden Land. Weitergehend gibt der Wert an, wie sehr sich die soziale Unterstützung auf die Bewertung des entsprechenden Landes auswirkt.

### 1.1.6 Healthy life expectancy

Dieses Feature beinhaltet den Wert, der angibt wie stark sich die Lebenserwartung in dem entsprechenden Land auf die Bewertung des Landes auswirkt.

### 1.1.7 Freedom to make life choices

Der Wert in diesem Feature gibt an wie stark sich die Möglichkeit eigene Lebensentscheidungen zu treffen auf die Beurteilung des entsprechenden Landes auswirkt.

### 1.1.8 Generosity

„Generosity“ gibt an, wie sehr sich die Großzügigkeit des entsprechenden Landes sich auf dessen Bewertung auswirkt.

### 1.1.9 Perception of corruption

In diesem Feature gibt der Wert an, wie stark in dem entsprechenden Land Korruption wahrgenommen wird und wie stark sich dieser Wert auf die Bewertung auswirkt.

# 2. Datenvorbearbeitung

Bevor die Daten für das Trainieren des Modells genutzt werden können, werden diese noch ein wenig angepasst.

Das Feature „*Country or region*” wird als eineindeutigen Index verwendet und ist damit leicht zu identifizieren und zu zuordnen.

# 3. Feature Engineering

Die heruntergeladenen Daten mussten nicht weiterbearbeitet werden. Die Feature können so verwendet werden wie sie existieren und eine neue Errechnung aus den in den Features gespeicherten Werten würde für die Vorhersage der Zufriedenheit eines Landes keinen Sinn ergeben.

Somit wurden in dem Notebook keine Feature Engineering betrieben.

# 4. Split des Datensatzes

Der vorbereitete Datensatz wird in drei Teile aufgeteilt. In Trainingsdaten, Validierungsdaten und Testdaten. Dabei wird die Funktion *train\_test\_splits()* von *scikit-learn* zweimal verwendet.

Im ersten Durchlauf wird der Datensatz in Trainingsdaten und Testdaten mit einem Verhältnis von 90% / 10% aufgeteilt. In diesem Durchlauf wurde mit dem Parameter *random\_state* dafür gesorgt, dass die Daten zufällig aufgeteilt worden sind, damit in den Trainingsdaten nicht nur die Länder sind, die eine niedrige Bewertung erhalten haben.

Im zweiten Durchlauf werden Trainingsdaten erneut in Trainingsdaten und Validierungsdaten mit einem Verhältnis von 75% / 25% aufgeteilt. Hier wurde nicht zufällig aufgeteilt, da die Daten bereits aus der ersten Aufteilung in einer zufälligen Anordnung waren.

## 4.1 Trainingsdaten

Die Trainingsdaten werden hierbei fürs Trainieren des Machine Learning Modells verwendet und bilden deswegen auch den größten Teil mit etwa 67% (105 Records).

## 4.2 Validierungsdaten

Die Validierungsdaten werden dazu verwendet das trainierte Modell zu überprüfen und zu optimieren und haben dabei den zweitgrößten Teil mit etwa 23% (35 Records).

## 4.3 Testdaten

Die Testdaten werden für Demonstration des Modells verwendet und haben deswegen den kleinsten Anteil mit etwa 10% (16 Records).

# 5. Das ausgewählte Machine Learning Modell

Für die Vorhersage der Bewertung der Zufriedenheit in einem Land wird eine Elastic-Net Regression verwendet.

## 5.1 Wie funktioniert das Elastic-Net Modell?

Das Elastic-Net Modell ist ein reguliertes Regressionsmodell. Es kombiniert dabei die L1-Norm und die L2-Norm der Lasso- und der Ridge Regression. Das Modell wählt dabei die Parameter zu selben Zeit, während andere Modelle (wie z.B. die Lasso Regression).

Das Modell hat dabei verschiedene Parameter, die für die Optimierung des Modells verwendet werden können. In diesem Projekt werden dafür die zwei Parameter alpha und l1\_ration verwendet, da diese hier die größte Auswirkung auf das Modell haben.

### 5.1.1 Der Parameter *alpha*

Der übergebene alpha-Wert ist die Konstante, mit der die Normen L1 und L2 multipliziert werden. Es gibt dabei an wir sehr das Modell „bestraft“ – bzw. reguliert – wird.

Der Standart-Wert für diesen Parameter ist 1.0, der verwendet wird, wenn man keinen anderen Wert übergibt.

### 5.1.2 Der Parameter *l1\_ratio*

Dieser Parameter gibt an welches Norm – bzw. in welcher Kombination die beiden Normen – verwendet werden. Der übergebene Wert kann nur in einem Wertebereich von einschließlich Null bis Eins sein.

Ist der übergebene Wert 0.0, so wird die L2-Norm für die Regulierung des Modells verwendet. Ist der übergebene Wert 1.0, so wird die L1-Norm verwendet. Bei einem übergebenen Wert zwischen Null und Eins wird eine dementsprechende Kombination der beiden Normen für die Regulierung des Modells verwendet.

## 5.2 Warum das Elastic-Net Modell?

Das ElasticNet Modell ist sehr gut für Vorhersagen geeignet bei deren Daten eine hohe Korrelation aufweisen. Das ist in dem hier ausgewählten Datensatz der Fall, da alle Feature für die Kalkulation des Zufriedenheitswertes verwendet werden.

# 6. Die ausgewählten Metriken

Für die Bewertung der Fehlerfreiheit des entsprechenden Modells werden vier unterschiedliche Metriken genutzt: *Mean Squared Error (MSE)*, *Root Mean Squared Error (RMSE)*, *Mean Absolute Error (MAE)* und *Mean Absolute Percentage Error (MAPE)*.

## 6.1 Mean Squared Error und Root Mean Squared Error

Der *Mean Squared Error* (oder auch *mittlere quadratische Abweichung*) berechnet sich aus der Summe der Abweichungen der einzelnen Vorhersagewerten von den tatsächlichen Werten ins Quadrat. Zuletzt wird die Summe durch die Anzahl der Werte geteilt. Er besagt den durchschnittlichen Abstand zwischen den geschätzen und den tatsächlichen Werten.

Der *Root Mean Squared Error* (oder auch Standartabweichung) berechnet sich aus der Wurzel der mittleren quadratischen Abweichung. Dieser Wert gibt die Standartabweichung der Regressionsgeraden und den (tatsächlichen) Datenpunkten an.

## 6.2 Mean Absolute Error

Der *Mean Absolute Error* (oder auch *mittlerer absoluter Fehler*) berechnet sich aus der Summe dem Betrag aller Abweichungen zwischen tatsächlichen und geschätzten Werten. Zuletzt wird die Summe durch die Anzahl der Werte geteiltgeteilt durch die Anzahl an Werten. Dieser Wert gibt die Höhe der Abweichung zwischen tatsächlichem und vorhergesagtem Wert an, ohne dabei die Richtung vorzugeben.

## 6.3 Mean Absolute Percentage Error

Der *Mean Absolute Percentage Error* (oder auch *mittlerer absoluter prozentualer Fehler*) ist eine der meistgenutzten Metriken, um Fehler in Vorhersagen zu messen und ist leicht zu interpretieren. Er berechnet sich aus der Summe des Betrags der Differenz der tatsächlichen Daten und der vorhergesagten Daten, geteilt durch die tatsächlichen Daten. Die Summe wird nochmal durch die Anzahl der Werte geteilt. Der Wert besagt die prozentuale Abweichung der vorhergesagten Daten von den tatsächlichen Daten.

# 7. Training des Machine Learning Modells

Das Model wird mit Hilfe der fit()-Funktion des Modells trainiert. Dafür werden die Trainigsdaten verwendet, die zum einen aus allen Featuren bestehen die zur Herleitung des Zufriedenheitswertes verwendet werden und zum anderen aus den dazugehörigen Zufriedenheitswerten.

# 8. Tuning der Hyperparameter

Um die herauszufinden mit welchen Parametern das Elastic-Net Modell das beste Ergebnis für die Vorhersage der verwendeten Daten liefert wird nacheinander ein Elastic-Net Modell erstellt, welches jedes Mal neue Parameter übergeben bekommt. Die Fehlerfreiheit jedes Modells wird mit den ausgewählten Metriken berechnet.

## 8.1 Die Werte der Parameter

Um herauszufinden welches der Modelle das Beste ist werden Modelle mit unterschiedlichen Werten für die Parameter *alpha* und *l1\_ratio* verwendet.

Für *alpha* wurden die Werte 1.0, 2.0, 3.0, 4.0, 4.5, 4.6, 4.7, 4.8, 4.9, 5.0 und 6.0 und für *l1\_ratio* wurden die Werte 0.0, 0.5 und 1.0 getestet.

Mit Hilfe von Schleifen konnte ein Modell mit jeder möglichen Kombination aus den Werten der zwei Parameter erstellt werden.

### 8.2 Der Einsatz von MLFLow

Mit der Hilfe von MLFlow werden das sowohl die Parameter des erstellten Modells, die dazu entsprechenden errechneten Metriken und das Modell selbst gespeichert. Über die grafische Benutzeroberfläche von MLFlow kann man die Fehlerfreiheit der verschiedenen Modelle vergleichen und herausfinden welches das Beste ist.

Das Beste der Modelle kann anschließend in das Demo-Notebook importiert werden, wo es dann die Testdaten vorhersagt.

Nach dem Parametertunen wurde das Modell mit den Parametern *alpha = 4.7* und *l1\_ratio = 1.0* als das insgesamt beste Modell identifiziert.

# 9. Evaluation mit den Testdaten

Für die Demonstration des Modells im zweiten Notebook (demo.ipynb) werden die Testdaten verwendet, die bei Aufteilen des Daten weggelegt worden sind. Diese Daten wurden dabei von dem Modell noch nie zuvor gesehen und sind ihm völlig fremd.

Nachdem das ausgewählte traineirte Modell importiert worden ist werden mit Hilfe der predict()-Funktion die Vorhersagen für die Zufriedenheitswerte der Testdaten gemacht.

Die vorhergesagten Daten erreichen dabei eine Genauogekeit von MSE = 0.04, RMSE = 0.2, MAE = 0.13 und MAPE = 2.65.

In einem Graphen lassen sich die Abweichungen genauer betrachten

Chart, line chart

Description automatically generated

Das Modell hat

# 10. Mögliche Schwachstellen und Verbesserungsmöglichkeiten