Data Exploration Project

Report

Vorgelegt am 15.04.2022

Fakultät: Wirtschaft
Studiengang: Wirtschaftsinformatik – Data Science
Kurs: WWI2020F
Vorlesung: Data Exploration Project
Dozent: Herr Nils Zerrer

Von Jasmin Noll

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

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.

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*”.
Das Feature “*Country or region*” beinhaltet dabei das Land – bzw. die Region – von der die restlichen Daten gesammelt wurden.
Das Feature “*Score”* gibt den Zufriedenheitswert des entsprechenden Landes/Region an. Dieses Feature wird hier als Zielfeature verwendet.
Die restlichen Features geben an wie sehr sich die entsprechenden Features auf den Zufriedenheitswert auswirken.

2. Datenvorbearbeitung

Bevor die Daten für das Trainieren des Modells genutzt werden können, werden die Daten auf ihre Vollständigkeit überprüft und angepasst.
Als erstes wird das Feature „*Country or region*” als eineindeutigen Index verwendet und ist damit leicht zu identifizieren und zu zuordnen. Dabei wurde zuvor überprüft, ob es Dopplungen der Länder/Regionen im Datensatz gibt. Dadurch gibt es nur noch Feature mit nummerischen Werten, die nicht weiter angepasst werden müssen.
Weitergehend wird überprüft, ob sich Null-Werte in dem Datensatz befinden. Dabei hat sich herausgestellt, dass es keine Null-Werte in diesem Datensatz existieren und keine weiteren Schritte eingeleitet werden müssen.

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

Die Validierungsdaten werden dazu verwendet das trainierte Modell zu überprüfen und zu optimieren und haben dabei den zweitgrößten Teil mit etwa 23% (35 Records).
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 zur selben Zeit, während andere Modelle (wie z.B. die Lasso Regression) dies nicht können.
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 der beiden Normen – 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 Elastic-Net 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* oder *RMSD)*, *Mean Absolute Error (MAE)* und *Mean Absolute Percentage Error (MAPE)*. Alle diese Metriken berechnen die Abweichungen zwischen Vorhergesagten Werten und den entsprechenden tatsächlichen Werten. Dementsprechend ist das Modell am besten, wenn die Metriken Null sind.

6.1 Mean Squared Error

Der *Mean Squared Error* (oder auch *mittlere quadratische Abweichung*) besagt den durchschnittlichen Abstand zwischen den geschätzten und den tatsächlichen Werten.

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

6.2 Root Mean Squared Error

Der *Root Mean Squared Error* (oder auch Standartabweichung) gibt die Standartabweichung der Regressionsgeraden und den (tatsächlichen) Datenpunkten an.

$$\mathrm{RMSD}(\hat{\theta}) = \sqrt{\mathrm{MSE}(\hat{\theta})}$$

6.3 Mean Absolute Error

Der *Mean Absolute Error* (oder auch *mittlerer absoluter Fehler*) gibt die Höhe der Abweichung zwischen tatsächlichem und vorhergesagtem Wert an, ohne dabei die Richtung vorzugeben.

$$\mathrm{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}$$

6.4 Mean Absolute Percentage Error

Der *Mean Absolute Percentage Error* (oder auch *mittlerer absoluter prozentualer Fehler*) besagt die prozentuale Abweichung der vorhergesagten Daten von den tatsächlichen Daten.

$$\mathrm{MAPE} = \frac{100\%}{n} \sum_{t=1}^{n} \left| \frac{A_t - F_t}{A_t} \right|$$

7. Training des Machine Learning Modells

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

8. Tuning der Hyperparameter

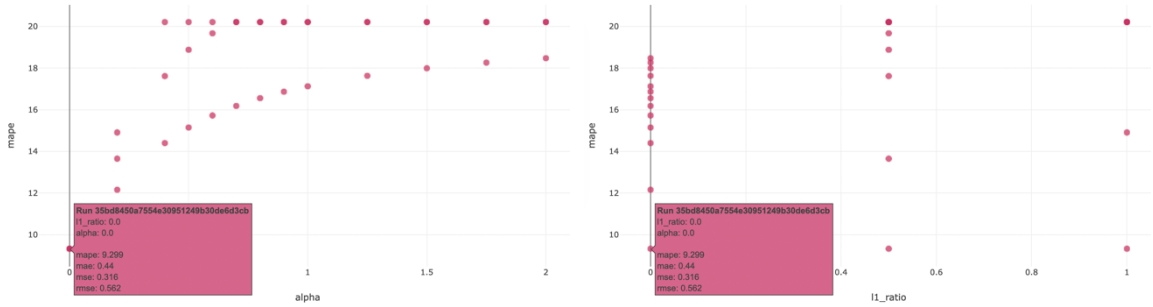
Um herauszufinden mit welchen Parametern das Elastic-Net Modell das beste Ergebnis für die Vorhersage der verwendeten Daten liefert, wird nacheinander ein Elastic-Net Modell erstellt, welches jedes Mal neue Parameter übergeben bekommt. Die Fehlerfreiheit jedes Modells wird mit den ausgewählten Metriken 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 0.0, 0.2, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.25, 1.5, 1.75 und 2.0 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

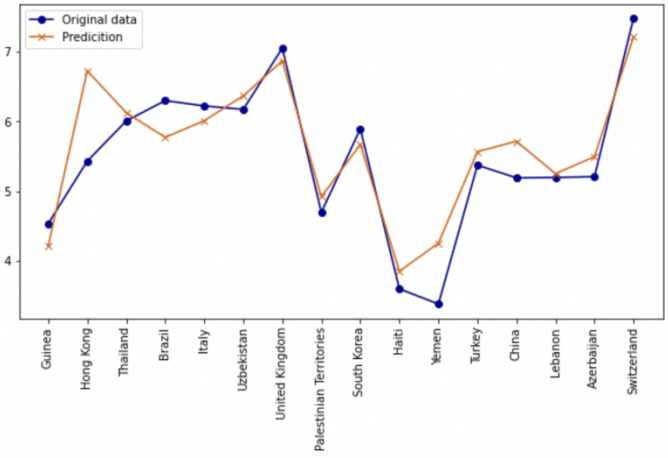
Mit der Hilfe von MLFlow werden das sowohl die Parameter des erstellten Modells, die dazu entsprechenden errechneten Metriken und das Modell selbst gespeichert. Über die grafische Benutzeroberfläche von MLFlow kann man die Fehlerfreiheit der verschiedenen Modelle vergleichen und herausfinden welches 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 = 0.0* und *l1_ratio = 0.0* als das insgesamt beste Modell identifiziert.



Das bedeutet, dass das Model nicht bestraft und die L2-Norm zur Regulierung verwendet wird.

9. Evaluation mit den Testdaten

Für die Demonstration des Modells im zweiten Notebook (*demo.ipynb* oder Google-Colab: https://colab.research.google.com/drive/1gRgP7eUHicIGm0nuqYa765ioiyw21Mcn#scrollTo=2VbVqhGX3c5t) 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 trainierte Modell importiert worden ist werden mit Hilfe der *predict()*-Funktion die Vorhersagen für die Zufriedenheitswerte der Testdaten gemacht.
Die vorhergesagten Daten erreichen dabei eine Genauigkeit von MSE = 0.22, RMSE = 0.47, MAE = 0.36 und MAPE = 7.24.
In einem Graphen lassen sich die Abweichungen genauer betrachten.



Die Schätzungen des Modells sind in den meisten Fällen recht nahe zu den tatsächlichen Werten. Auffallende Abweichungen sind bei den Werten für *Hong Kong*, *Brasilien*, *Jemen* und *China*. Hier besteht die größte Differenz zu den originalen Datenpunkten.
Die restlichen Daten sind allerdings ziemlich genau – mit nur kleinen Abweichungen – getroffen worden. Grundsätzlich kann man sich an den Werten, die das Modell vorhersagt, für eine Einordnung orientieren.

10. Mögliche Schwachstellen und Verbesserungsmöglichkeiten

Der verwendete Datensatz ist kein großer Datensatz. Dadurch stehen nur wenige Daten zum Trainieren des Modells zur Verfügung. Mit einer größeren Anzahl an Daten kann das Modell besser lernen Muster zu erkennen und Vorhersagen mit einer größeren Genauigkeit treffen. Daten über mehrere Jahre wären dafür sehr hilfreich, allerdings besteht bei den vorhanden Daten keine Konsistenz zum Aufbau der Daten, weshalb nur die aktuellsten Daten – die aus dem Jahr 2019 – verwendet worden sind.
Weitergehend könnte es auch der Fall sein, dass es im Datensatz Fehler zur Berechnung des Zufriedenheitswertes anhand der übrigen Daten gibt. Durch solch einen Fehler könnten Vorhersagen nicht genau getroffen werden.