InformatiCup - Dokumentation

Tilman Hinnerichs, Tobias John

20. Januar 2018

Zusammenfassung

Im Folgenden soll unsere Lösung der Aufgaben des 13. InformatiCup 2018 vorgestellt werden. Dabei möchten wir unsere Annahmen zur Aufgabenstellung, unsere Lösung für beide Probleme und deren Bewertung, und einen Ausblick unserer Lösung beschreiben und erklären.

1 Annahmen zur Aufgabenstellung und Umformungen der Eingabedaten

Für Umsetzung unserer Lösung mussten wir dafür sorgen, dass die Werte nicht nur eingelesen, sondern auch so umgeformt werden, dass diese bestimmte Kriterien erfüllen.

1.1 Umformung der Tankstellendaten

Die Datei der Tankstellendaten stellt sehr viele unterschiedliche Daten zur Verfügung. Die reichen von der ID der Tankstelle, über die Postleitzahl zu den Koordinaten der Tankstelle. Für unser Verfahren ist sind folgende Kriterien wichtig:

1. **Individualität** der Tankstellen um diese eindeutig zuordnen und um über diese in einer Liste iterieren zu können
2. **Örtliche Zuordnung** der Tankstellen um zwischen ihnen Abstände zu errechnen

Um eventuell später Zusammenhänge einzubeziehen oder wiederzuerkennen, haben wir zusätzlich die Marke der einzelnen Tankstellen mit einbezogen. Dies bildet aber zur Erfüllung der obigen Kriterien keinen Mehrwert.
So werden bei der Einbeziehung der Tankstellendaten nur deren ID, Marke und Koordinaten aus den Tankstellendaten aus oben genannten Gründen herausgelesen.

1.2 Umformung der historischen Benzinpreisdaten

Die Daten, die wohl am wichtigsten für die Voraussage der Benzinpreise selbst sind, sind die historischen Tankstellendaten. Bei diesen stehen Tausende von Daten pro Tankstelle zur Verfügung. Pro Änderung des Benzinpreises stehen dabei das Datum mit der Uhrzeit mit einer Genauigkeit bis auf die Sekunde genau und der zugehörige neue E5-Benzinpreis zur Verfügung. Kriterien für die Umformung dieser Daten sind die folgenden:

1. **Vergleichbarkeit** der Datensätze um mit ihnen eine Einteilung in bestimmte Kategorien vorzunehmen
2. **Kontinuität** der Daten um eine Vergleichbarkeit der Daten bestimmter gleicher Intervalle zu erreichen und um diese in Algorithmen einfüttern zu können
3. **Diskretisierung** der gemessenen Zeitpunkte
4. **Handhabbare Menge** an Daten um mit ihnen noch in annehmbarer Zeit rechnen zu können, aber ebenfalls noch so viel, dass nicht zu viel Information verloren geht

Um eine Vergleichbarkeit der Datensätze zu gewährleisten mussten wir die Kontinuität der Daten erreichen. Dazu rundeten wir jeden der Änderungszeitpunkte auf seine Stunde herunter und rechneten dabei ebenfalls die zugehörige Zeitverschiebung mit ein. Ebenfalls reicherten wir die Daten mit den Preisen zwischen diesen Änderungszeitpunkten an. Was auf den ersten Blick recht trivial wirkt, nimmt bei weiteren Algorithmen sehr viel Arbeit ab. So beträgt beispielsweise der Benzinpreise nach einer Änderung auf ein bestimmtes Niveau bis zu seiner nächsten Änderung diesen Wert bei. Durch diese Anreicherung besteht ein jeder Tag von Werten in der Geschichte einer Tankstelle aus 24 Werten. Dies bietet ebenfalls eine wunderbare Unabhängigkeit von schnell schwankenden Preisen innerhalb eines Tages und eben die oben gewünschte Vergleichbarkeit, da nun die Werte eines Tages mit 24 Werten dargestellt werden können. Damit ist zugleich die Kontinuität durch die gleichen Intervalle zwischen den Messpunkten und die Diskretisierung durch 24 feste Punkte innerhalb der sonst linearen Zeit gegeben.
Abschließend ist aus unserer Sicht das dilemmaartige vierte Kriterium der handhabbaren Menge an Daten gewährleistet. So sind durch die Einteilung noch eine Menge von $24 * 365 * 3 = 26.280$ Werten (bei ca. 3 betrachteten Jahren der historischen Tankstellenwerte) vorhanden, was eine ausreichende Genauigkeit für zukünftige Voraussagen mit unserer Methode bietet und eine für Computer komfortable Anzahl an Werte bildet.

1.3 Umformung der Routendaten

Die Routendaten sind vor allem für die Berechnung der besten Route von Bedeutung. Diese enthalten neben der maximalen Tankkapazität des Fahrzeugs auch die Ankunftszeiten an den bestimmten Tankstellen, sowie deren ID. Um hier die Kausalität der bisherigen Daten weiterzuführen, diskretisieren wir wieder die Daten auf die volle Stunde und verwerfen für unser Modell zu spezifische Daten wie Minuten und Sekunden. Weitere Annahmen müssen dabei für die Routendaten nicht getätigt werden, um sie in Konformität zu bringen.

1.4 Umformung der Preisvorhersagedaten

Die getätigten Annahmen zu den Preisvorhersagedaten sind ähnlichen zu den unter „Umformung der Routendaten“ getätigten Annahmen. So wird wieder sowohl der Zeitpunkt, welcher als zuletzt bekannt angenommen werden soll, als auch der Zeitpunkt, bis zu welchem die Vorhersage reichen soll, auf die jeweilige Stunde gerundet, um in das oben angerissene Datenmodell zu passen.

2 Algorithmenidee

2.1 Lösung der Benzinpreis-Vorhersage-Problems

2.1.1 Ansatz

Welche Algorithmen wurde hier benutzt uns was versprechen wir uns davon? Warum haben wir das ganze in Intervalle unterteilt? Gehen wir auf etwaige Sonderdaten wie in der Aufgabenstellung genannt ein (Schulferien, Adresse,)

Annahmen

Unsere Vorhersagemodell beruht auf zwei simplen Annahmen:

1. Die Preise folgen **sich wiederholenden Mustern**, d.h. die aktuelle Preisentwicklung wird sich fortführen, wie es ähnliche Entwicklungen in der Vergangenheit getan haben. Folgte z.B. in der Vergangenheit auf drei Tage starken Preisstiegs immer ein Abfall, so wird dies in Zukunft auch ähnlich passieren.

2. Tankstellen, die über den Jahresverlauf eine ähnliche Preisentwicklung aufweisen (z.B. wegen gleicher Ferien, d.h. örtlich enger Lage), haben auch im Kleinen (d.h. Tagesverlauf) eine ähnliche Preisentwicklung.

Die erste Annahme sollte unbestreitbar sein. Ohne sie wäre ein Vorhersagemodell für Benzinpreise auch nicht möglich.
Die zweite Annahme ist schon diskussionswürdiger. Es gibt keine mathematische Begründung, warum diese Korrelation bestehen sollte. Jedoch wurde unsere Vorhersage deutlich besser, wenn der Algorithmus so modifiziert wurde, dass er Annahme zwei berücksichtigt.
Die zweite Annahme beachtend sortiert unser Algorithmus die Tankstellen zunächst Äquivalenzklassen ein. In einer Klasse landen dann sich gegenseitig ähnliche Tankstellen. Für jede der Klassen wird dann ein eigene Vorhersagemodell erstellt, mit dem die Vorhersagen für die entsprechenden Tankstellen durchgeführt werden.

Einteilung in Äquivalenzklassen

Die Einteilung in Klassen kann mit Hilfe verschiedener Parameter (z.B. Postleitzahl, Bundesland, Marke,...) erfolgen. Diese Kriterien garantieren jedoch mitnichten eine Korrelation der Benzinpreise. Darum entschieden wir uns die Tankstellen **ausschließlich** mit Hilfe ihrer historischen Benzinpreise zu sortieren. Dies garantiert die Ordnung nach Benzinpreisen. Natürlich werden dabei, dank Schulferien, die Tankstellen automatisch auch nach Lage sortiert. Dementsprechend ist eine Hinzufügung weiterer Daten für unser Vorhersagemodell nicht hilfreich.
Zunächst formten wir die gegebenen Benzinpreise jeder Tankstelle in einen Vektor um, der die Benzinpreisentwicklung wieder spiegelt. Um die Sortierung zu realisieren, verwendeten wir **Selforganising-Feature-Maps** (SOFMs). Diese sind ein gut studiertes Werkzeug der Neuroinformatik und eignen sich hervorragend für die Einordnung von hochdimensionalen Vektoren in ähnliche Klassen. Dazu wird ein Netz (hier zweidimensional) von Neuronen durch Training so im hochdimensionalen Werteraum plaziert, dass es möglichst gut die gegebenen Vektoren abbildet. Das Netz passt sich also der Verteilung der gegebenen Vektoren an. (Für eine detailliertere Beschreibung des Algorithmus siehe [1])
Nach dem Training kann für jede Tankstelle das Neuron ermittelt werden, welches die kleinste euklidische Distanz zum Vektor der Tankstelle aufweist. Dieses wird als best-matching-Neuron

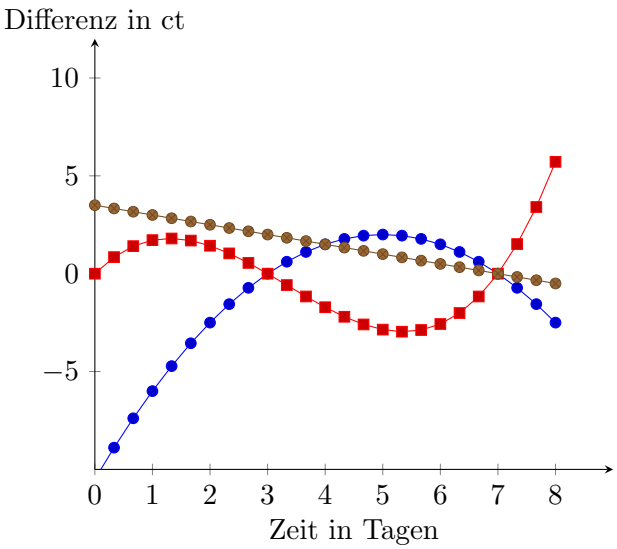

Abbildung 1: Beispielhafter Presiverlauf über acht Tage

bezeichnet. Tankstellen mit dem selben best-matching-Neuron werden eine Klasse gesteckt. Damit hat diese SOFM unsere Tankstellen in Äquivalenzklassen eingeteilt. Im nächsten Schritt wurde für jede Klasse ein eigenes Vorhersagemodell entwickelt und angewendet.

Einzelnes Vorhersagemodell

Obwohl die Menge der zur Verfügung stehenden Trainingsdaten durch die Einteilung in Äquivalenzklassen bereits drastisch reduziert wurde, ist die Datenmenge immer noch gigantisch groß. Darum verwendeten wir auch hier wieder ein neuronales Modell, welches zunächst auf den gegebenen Daten trainiert wurde und anschließend (schnell) für gegebene Daten eine Vorhersage treffen kann. Unser Modell trifft dabei aus dem Preisverlauf der vergangenen Woche eine Vorhersage für den nächsten Tag. Durch iterative Anwendung der Vorhersage kann unser Modell so beliebig weit in der Zukunft die Benzinpreise vorher sagen.
Da für unsere Anwendung passend, wählten wir wieder SOFMs als Modell aus.

Die zentrale Entscheidung war nun, wie die gegebenen Benzinpreise vorverarbeitet werden müssen, so dass sie von der SOFM benutzt werden können. Wir entschieden uns mit der SOFM acht Tage an Benzinpreisen zu verarbeiten, mit je einem Wert pro Stunde. Dies ergibt 192-dimensionale Vektoren. Der erste Preis ist auch an einem Tag um 00:00 Uhr entnommen worden, d.h. die Datensätze beginnen immer mit einem Tagesbeginn. Dies hat den Vorteil, dass das Netz Preisverläufe im Tagesverlauf besser abschätzen kann.
Da uns nur die Entwicklung, nicht aber die Höhe des Preises interessiert, wird von allen Datenpunkten der Preis des siebten Tages 23:00 Uhr abgezogen. Dieser Wert ist damit in allen Datensätzen die verarbeitet werden null. Damit ist es möglich auch Preisevorhersagen zu treffen, obwohl die entsprechenden Preise noch nicht vor kamen. D.h. ein heute trainiertes Modell kann auch in ferner Zukunft aus einer Woche an gegebenen Daten mit beliebig hohen Benzinpreisen eine Vorhersage berechnen. Wir benötigen für eine Vorhersage (nach Training der SOFM) also nicht die (halbwegs) aktuellen Daten aller Tankstellen, sondern es reicht uns ein

winziger Bruchteil an aktuellen Daten der einzelnen Tankstelle, um ihre Preise vorherzusagen. Das bedeutet auch, dass eine einzelne Vorhersage sehr schnell ausgewertet werden kann.
In Abbildung 1 sind beispielhaft drei Datensätze dargestellt, mit denen unsere SOFM trainiert werden könnte. Man sieht deutlich, dass sie einen Punkt am Ende des siebten Tages mit dem Funktionswert null gemeinsam haben.
Um die SOFM zu trainieren, benötigen wir Trainingsdaten, an denen sich die Neuronen dann ausrichten. Dafür wählen wir aus den historischen Daten der Tankstellen zufällige Datenabschnitte aus. Bei hinreichend großer Anzahl an Datenabschnitten entsteht so ein Datensatz der repräsentativ für die verschiedenen Benzinpreisverläufe ist.

Preisvorhersage
x

Wir sollen das ganze an folgenden vergleichbaren Schritten herleiten und erklären:

1. Zusätzliche Informationen?
2. Diskutieren der Prognoseergebniss (8GB RAM)

Und zusätzlich:

> Überlegen Sie im nächsten Schritt, welche zusätzlichen Informationen wie zum Beispiel die Wochentage, Ferienzeiten oder Verkehrsinformationen für Ihre Benzinpreisvorhersagen sinnvoll sein könnten. Erweitern Sie Ihre Vorhersagemodelle entsprechend und dokumentieren Sie Ihre Ergebnisse.

2.1.2 Umsetzung der Lösung

Wie haben wir diese Idee umgesetzt? Warum haben wir wie viele Dimensionen an die SOM/SOFM vergeben? Warum halten wir das für sinnvoll?

2.1.3 Bewertung der Lösung

Hier könnte Ihr Diagramm stehen.
Wie gut ist das was wir da gebastelt haben?
Laufzeit betrachten: Parallelität, Coreanzahl, Server? (mit GPU) –¿ Forecast dann schnell

> Diskutieren Sie die Güte Ihrer erzielten Prognoseergebnisse für den geforederten Vorhersagezeitraum (d.h. bis zu einem Monat in die Zukunft). Verwenden Sie dazu geeignete Maßzahlen für die Güte Ihrer Vorhersagemodelle.

2.2 Lösung des Routenproblems

2.2.1 Ansatz der Lösung

Für die Lösung des Routenproblems wurde der in der Aufgabenstellung beschriebene Algorithmus „To fill or not to fill“ benutzt. Der Ansatz allein bedarf deswegen keiner weiteren Erläuterung, wobei unsere Umsetzung hingegen erklärenswert ist.

Das sehe ich anders. Es geht sicherlich auch darum, aus dem Paper den Algorithmus herauszulesen, d.h . wissenschaftliche Artikel zu verstehen. Wir sollten kurz zeigen, dass uns das gelungen ist. Außerdem muss hier unbedingt (!!!) noch die Quelle genannt werden.

2.2.2 Umsetzung der Lösung

Das Problem wurde dafür in mehrere Unterprobleme unterteilt, die sich wunderbar in einer Softwarelösung umsetzen lassen. So wurde dieses Problem in die Teilprobleme der Entfernungsberechnung, Preisfindung für eine spezifische Tankstelle auf der Route und den Algorithmus selbst untergliedert.
Für die Entfernungsberechnung wurde dabei die Formel für die Entfernung auf Großkreisen verwendet. Um von einer Tankstelle zu einer anderen Tankstelle zu fahren, wobei sich strikt an die vorgegebene Route gehalten werden muss wurde diese Entfernungsberechnung rekursiv aufgebaut. Eine Strecke von Knoten A zu Knoten B erfolgt demnach über die Berechnung und Summierung der Einzelstrecken zu den Knoten zwischen den A und B. Andere Methoden wie das direkte Fahren von A zu B, wenn dies der aktuelle Benzinstand zulässt, würde bei Routen wie der vorgegebenen Bertha-Benz-Memorial-Route, welche eine Rundreise darstellt, zu unerwünschten Effekten führen.
Für das Auffinden der bereits errechnete Preise müssen diese bereits berechnet vorliegen.
Der Algorithmus selbst bildet dabei das Kernstück der Tankstrategie und wurde wie in dem Paper gegeben umgesetzt. Dafür müssen die folgenden Teilprobleme wie im Kapitel der Implementierung beschrieben umgesetzt werden:

1. **Die Next-Funktion**, welche wie beschrieben dafür sorgt den billigsten nächsten Knoten zu finden.

2. **Die Previous-Funktion**, welche den billigsten zurückliegenden Knoten liefert. Falls keiner billiger sein sollte, wird der Startknoten selbst ausgegeben.

3. **Finden aller Breakpoints**, also aller Knoten die keinen Knoten hinter sich in Reichweite haben, sodass dieser einen billigeren Preis liefert.

4. **Fahren zum nächsten Knoten**, also das schließliche Weiterfahren, mit Berechnung der Verbrauchs und Berechnung der nachzutankenden Menge.

Wobei die beschreibenden Formeln in der Beschreibung des Algorithmus zu finden sind.

3 Implementierung und Umsetzung der Lösung

Hier könnte Ihr Klassendiagramm stehen. Entwurf und Struktur der Lösung

3.1 Klasse GasStation

Welche Aufgaben hat die Klasse zu erfüllen? Was sind die wichtigsten Funktionen und was tun diese? Wie werden die Datenstrukturen befüllt und warum auf diese Weise?

Welche der vielen, vorhandenen Daten benutzen wir überhaupt? Warum können wir den Rest verwerfen?

3.2 Klasse Route

Welche Aufgaben hat die Klasse zu erfüllen? Was sind die wichtigsten Funktionen und was tun diese? Wie werden die Datenstrukturen befüllt und warum auf diese Weise?

3.3 Klasse PrizeForecast

Welche Aufgaben hat die Klasse zu erfüllen? Was sind die wichtigsten Funktionen und was tun diese? Wie werden die Datenstrukturen befüllt und warum auf diese Weise?

3.4 Klasse Supervisor

Welche Aufgaben hat die Klasse zu erfüllen? Was sind die wichtigsten Funktionen und was tun diese? Wie werden die Datenstrukturen befüllt und warum auf diese Weise?

3.5 Klasse Model

Welche Aufgaben hat die Klasse zu erfüllen? Was sind die wichtigsten Funktionen und was tun diese? Wie werden die Datenstrukturen befüllt und warum auf diese Weise?

3.6 Klasse Strategy

Welche Aufgaben hat die Klasse zu erfüllen? Was sind die wichtigsten Funktionen und was tun diese? Wie werden die Datenstrukturen befüllt und warum auf diese Weise?

4 Ausblick und Erweiterungen

InformatiCup sagt, dass man tolle Erweiterungen einfügen könnte. Ein kleiner Auszug:

> Falls nach der erfolgreichen Implementierung der Grundanforderungen noch Zeit für Erweiterungen bleibt, seien Sie kreativ, zum Beispiel mit einer mobilen App für echte Benzinpreisvorhersagen unterwegs. Oder passen Sie Ihre Softwareanwendung für die Ausführung in der Cloud für eine hohe Performanz, Skalierbarkeit und Verfügbarkeit an. Sie möchten nicht mit einem durchschnittlichen Kraftstoffverbrauch rechnen? Integrieren sie mögliche Spezialsoftware für die Simulation des Benzinverbrauchs. Oder verwenden Sie Daten aus Online-Kalendern als Grundlage für Ihre Fahrzeugrouten. Seien Sie kreativ!

Uns weiterhin:

> ... Ausblick: Lässt sich Ihr Verfahren vielleicht auf auf Ladevorgänge in der Elektromobilität oder für ganz andere Aufgabenstellungen anwenden?

Literatur

[1] Wikipedia, Die freie Enzyklopädie (Hrsg.): Selbstorganisierende Karte; `https://de.wikipedia.org/w/index.php?title=Selbstorganisierende_Karte&oldid=172760713`, Zugriff 20.01.2018

[2] Yurii Shevchuk: NeuPy Home; http://neupy.com/pages/home.html, Zugriff 20.01.2018

[3] S. Khuller, A. Malekian und J. án Mestre: Fill or not to Fill: The Gas Station Problem; http://www.cs.umd.edu/projects/gas/gas-station.pdf, Zugriff 20.01.2018