InformatiCup 2018 - Benzlim

Franck Awounang Nekdem, **Gerald Wiese**, **Amin Akbariazirani** und **Lea Evers**

Leibniz Universität Hannover

InformatiCup 2018 - Benzlim
Einführung
Analyse
Ansatz
Training
Vorhersage
Klassifizierung
Vorhersage
Prädiktionen
Korrektor
Routing
Ergebnisse
Auswertung
Vorhersage
Bekannte Probleme
Abschluss
Ausblick
Danksagung

Einführung

Die Tankstrategie ist ein wichtiger Bestandteil jeder Reise. Wann, an welcher Tankstelle und wie viel tanken zu müssen um am günstigsten und effizientesten ans Ziel zu gelangen macht auf lange Sicht einen großen finanziellen Unterschied. Benzlim ist eine auf Python basierte Software-Lösung, die Verbraucher*innen und Entwickler*innen nutzen können, um Benzinpreise vorherzusagen und die effizienteste Tankstrategie zu erstellen.

Analyse

- **Benzinpreisentwicklung**

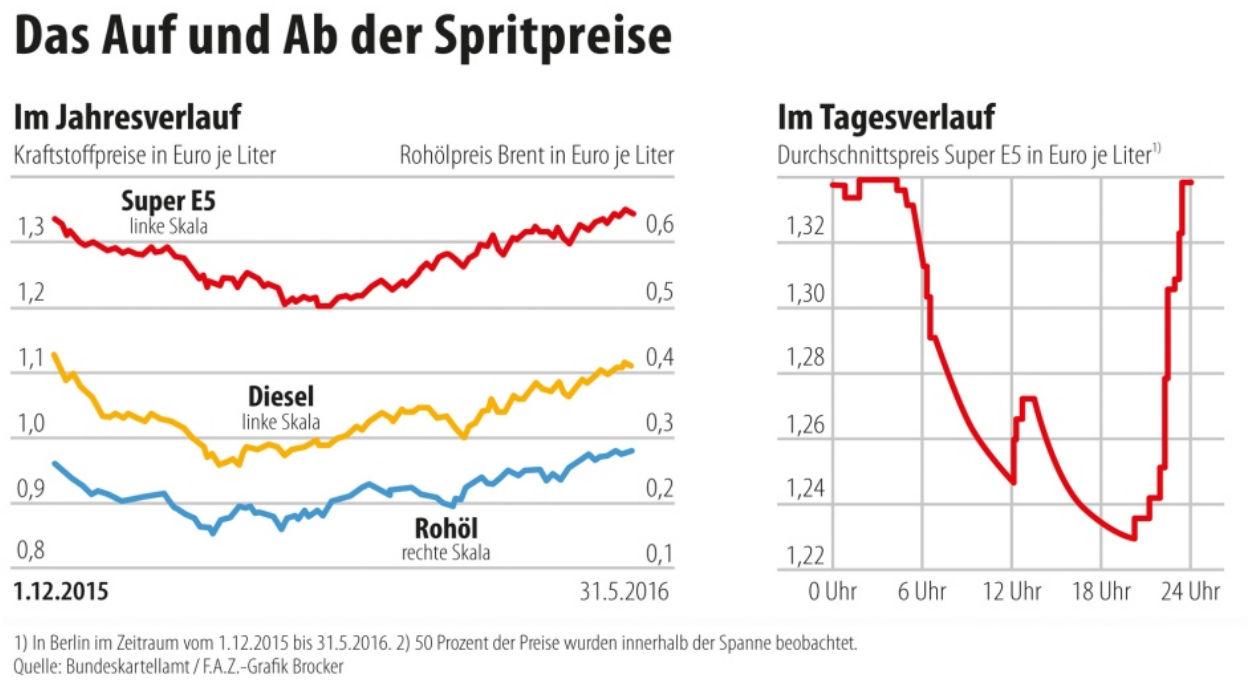

Bild 1. *Quelle:* Frankfurt Allgemeine Zeitung

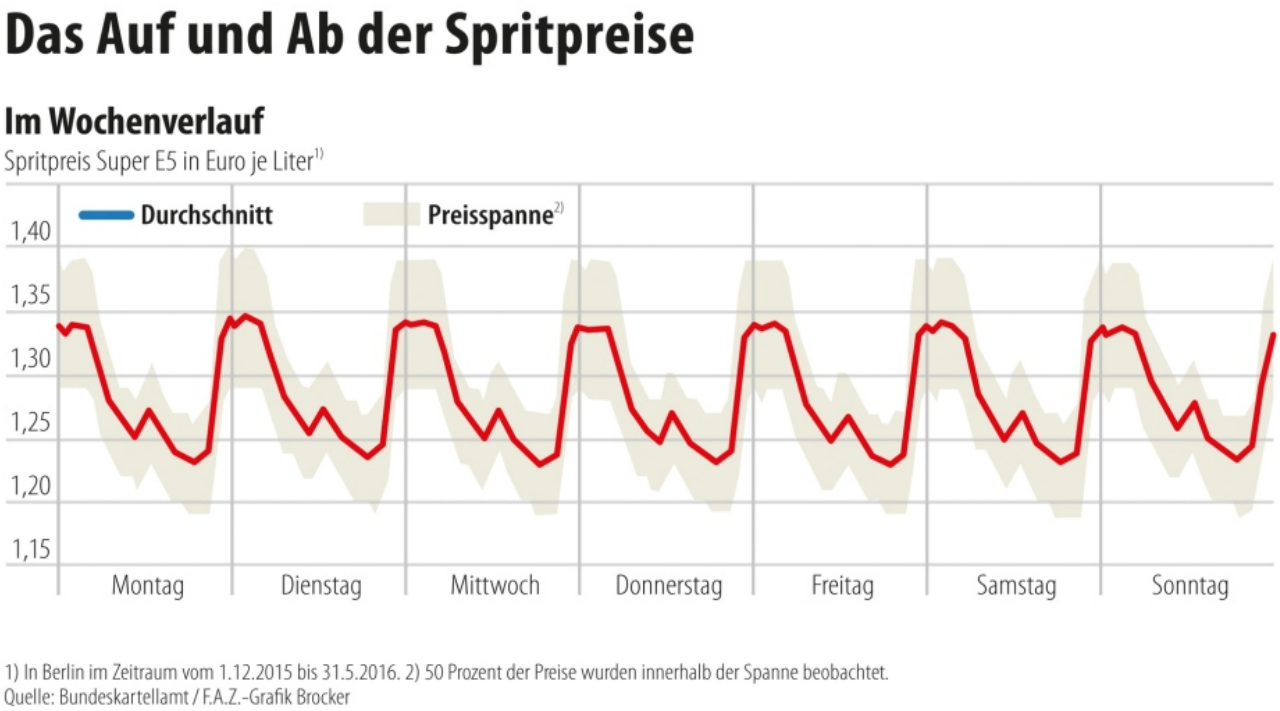

Bild 2. *Quelle:* Frankfurt Allgemeine Zeitung

Benzinpreise entwickeln sich periodisch täglich und jährlich, wie aus Bild 1., Bild 2. und mtsk_dritte_jahr zu erfahren ist. Und erfahren bist zu ca. 100 cents Preisunterschied am Tag.

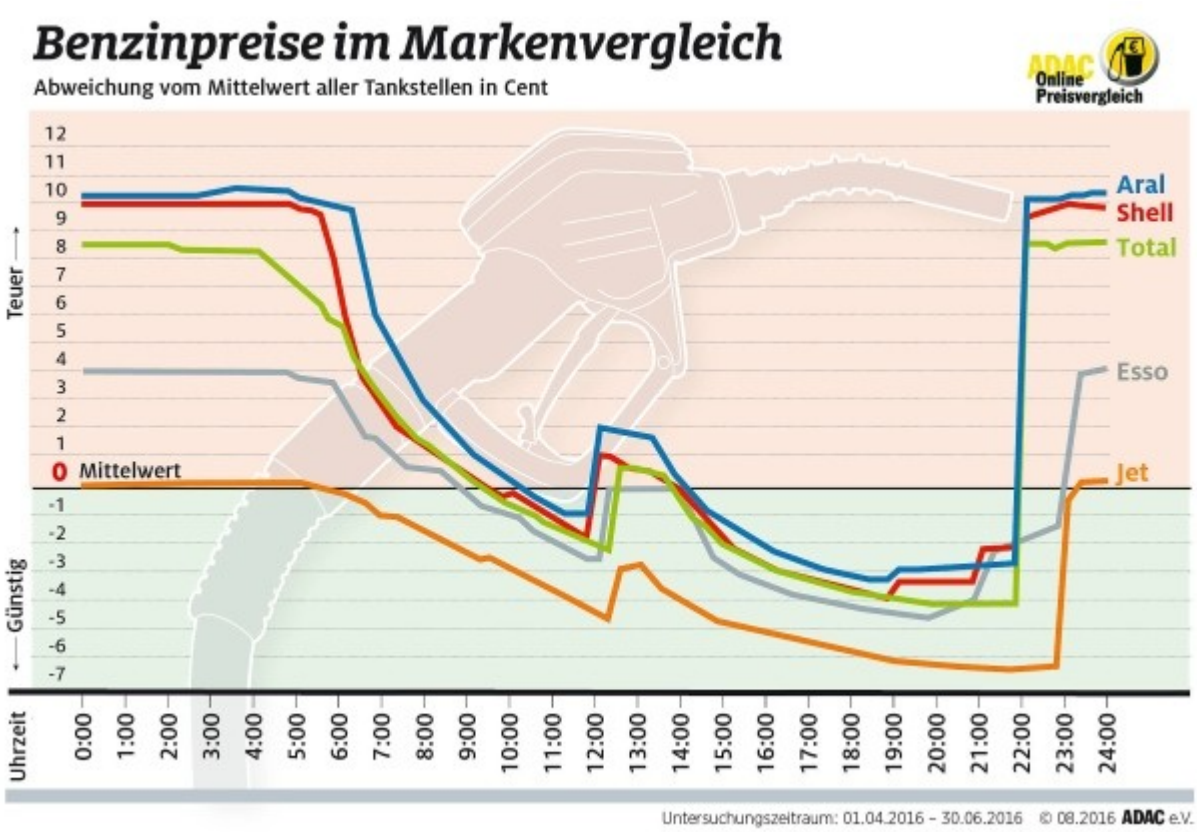

Bild 3. *Quelle:* [Frankfurt Allgemeine Zeitung]

Benzinpreisänderungen am Tag sind unabhängig von der Marke Bild 3. [mtsk_dritte_jahr].

Ansatz

Der ausgewählte Lösungsweg basiert darauf, dass die Benzinpreise stärker von der Marke als vom Ort abhängen. Um die Preise vorhersagen zu können, werden die durchschnittlichen Benzinpreise in bestimmten Zeitspannen (jährlich, monatlich, wöchentlich, täglich, stündlich, minütlich) berechnet. Die erzeugten Daten werden zu einem [Extrapolator] übergeben, der einen Prädiktor für die Differenz zwischen der jeweiligen Zeiteinheit und den höheren Zeiteinheiten erzeugt. Der grundlegende Prädiktor summiert die durchschnittlichen jährlichen Prädiktionen mit den monatlichen, wöchentlichen, täglichen, stündlichen und minütlichen Prädiktionen auf und erzeugt die Vorhersage.

Training

Für die weitere Verarbeitung werden die Daten gereinigt und optimal gespeichert. Um auf die Daten optimal zugreifen zu können, werden in der Trainingsphase die folgenden Schritte durchgeführt:

1. Eine lokale Datenbank mit Stationinformationen wird erzeugt
2. Die Stationinformationen werden um die Verfügbarkeit der Preise, sowie das Datum des ersten gemeldeten Preises erweitert.

Vorhersage

Klassifizierung

"S" ist die Menge aller bekannten Stationen und "S_p" ist die Menge aller Stationen sowie die dazugehörigen Preisinformationen. Die Klassifizierung gibt für eine Station "s" in "S" die passendste Station "s_p" in "S_p" aus und erfolgt wie im Bild 4. beschrieben.

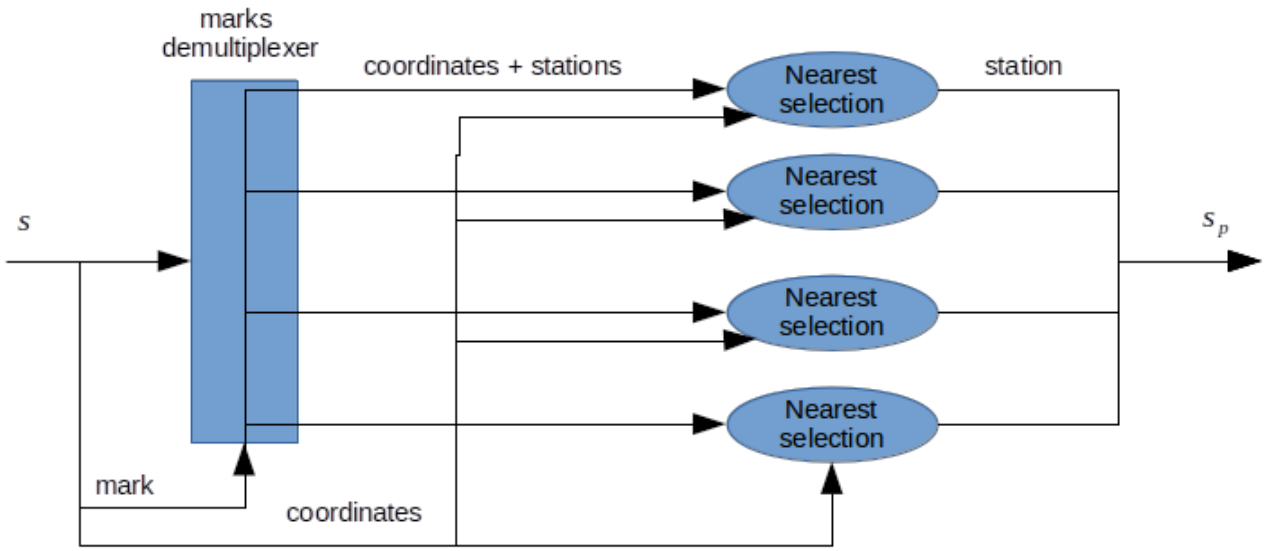

Bild 4.

Vorhersage

Prädiktionen

Pro Vorhersage wird ein Prädiktor *P* trainiert.

- Es werden Preise selektiert, die in dem gleichen Stundenzeitlot sind, wie der Zeitstempel für die Prädiktion
- Der Prädiktor besteht aus 6 Subprädiktoren
- Seien *yearly_avg*, *monthly_avg*, *weekly_avg*, *daily_avg*, *hourly_avg* und *min_avg* jeweils die jährlichen, monalichen, wochentlichen, täglichen und stündlichen durchschnittlichen Preise.
- Seien *monthly_rel*, *weekly_rel*, *daily_rel*, *hourly_rel* und *min_rel* die Unterschiede zwischen den jeweils monatlichen, wochentlichen, täglichen und stündlichen durchschnittlichen Preisen und den durchschnittlichen Preisen der höheren Zeiteinheit.
- Alle **_rel* werden zu einem Extrapolator übergeben, der einen Prädiktor für den Unterschied zwischen der jeweiligen Zeiteinheit und der höheren erzeugt.
- *yearly_avg* wird zu einem Extrapolator übergeben, der einen Prädiktor für den jährlichen durchschnittlichen Preis erzeugt. Jede *rel Tabelle berechnet sich aus den Unterschieden zwischen dem passenden **avg* und der Summe der Prädiktionen der höheren Zeiteinheiten.
- Alle **_rel* werden zu einem Extrapolator übergeben, der einen Prädiktor für den Unterschied zwischen der jeweiligen Zeiteinheit und der höheren erzeugt.
- Der grundlegende Prädiktor summiert die durchschnittlichen jährlichen Prädiktionen mit den monatlichen, wöchentlichen, täglichen, stündlichen und minütlichen Prädiktionen auf und erzeugt die Vorhersage.

- Es wird der Durchschnitt der selektierten Preise für die Vorhersage berechnet. Dieser wird als prädizierter Wert benutzt, falls den tatsächlichen prädizierten Wert eine Abweichung von 20% zu ihm weist. Somit ist der prädizierte Preis p_{p1} vom Prädiktor P_1 erzeugt.
- Als Zusatz wird *untrust = std(prices) / mean(selected_prices)* wo *std* die Standardabweichung ist und *avg* die Durchschnittfunktion berechnet. *untrust* gibt die Unsicherheit des Prädiktors an.

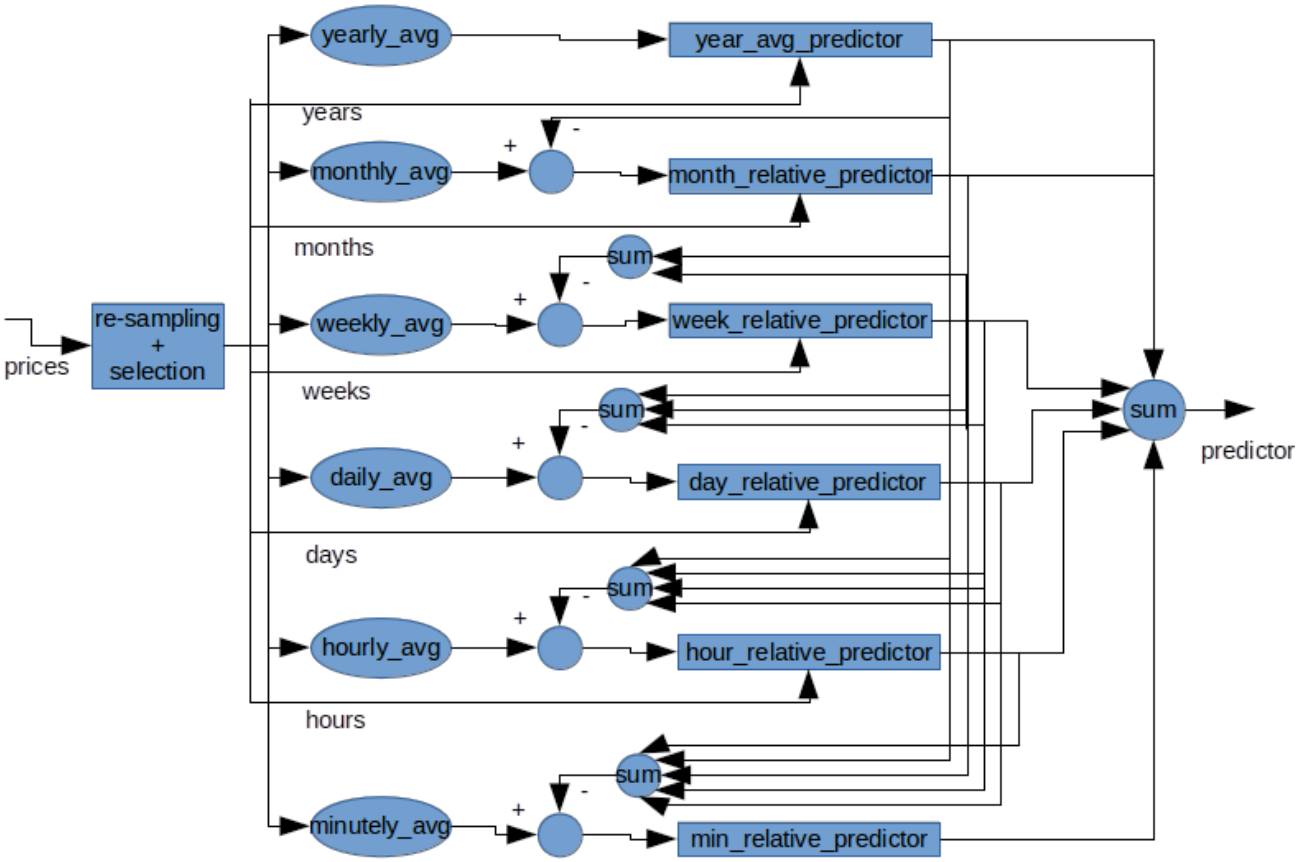

Bild. 5

Korrektor

- Ein zweiter Prädiktor *P2* mit nur einem Level wird trainiert. Er präzidiert einen Preis basierend auf der Anzahl der Nanosekunden in einem Zeitsstempel. Dieser Prädiktor verfügt eben über Autokorrektur und generiert zusätzlich zu dem prädizierten Preis p_{p2} eine Unsicherheit *untrust2* , die ausgibt wie unsicher der Prädiktor ist.
- Seien *trust1 = 1-untrust1* und *trust2 = 1 - untrust2*. Der endgültigte prädizierte Preis ist : p_p= *(trust1*p_{p1} + trust2p_{p2}*) / (trust1 + trust2)*

Routing

Basierend auf der Entfernung bis zur nächsten günstigsten Tankstelle und der Tankkapazität des Autos wird die richtige Strecke und die jeweils zu tankende Menge berechnet. Dabei können verschiedene Werte dynamisch verändert werden:

- *tolerance_km* gibt an, für wieviele Kilometer mehr als bis zur nächsten ausgewählten Tankstelle getankt wird, falls der Benzinverbrauch höher als erwartet ist.

- *fuel_surplus* gibt an, mit wieviel Benzin das Ziel erreicht werden soll.
- *tolerance_amount* und *tolerance_price* geben an, mit welcher Toleranz Tankstellen ausgelassen werden können, um nicht zu oft zu halten. Z.B. würden mit tolerance_amount = 10 und tolerance_price = 0.10 diejenigen Tankstellen übersprungen werden, bei denen weniger als 10 Liter getankt werden, falls die nächste oder letzte Tankstelle höchstens 10 Cent teurer ist.

Ergebnisse

Auswertung

Bei der Auswertung liegt der Fokus auf der Vorhersage der Preise. Ausgewertet sind sowohl Stationen mit verfügbaren Preisinformationen als auch Stationen die keine Daten zu deren Preisen zur Verfügung gestellt haben.

Vorhersage

Ausgewertet werden erst Prädiktionen, bei denen das Training mit tatsächlichen Preisen der Station durchgeführt wurde und anschließend Prädiktionen, bei denen das Training mit Preisen einer Ersatzstation durchgeführt wurde.

- Vorhersagen mit verfügbaren Preisen

  Wir haben 1000 Stationen mit verfügbaren Preisinformationen ausgewählt und für jede dieser Stationen ein Zufallsdatum erzeugt. Mit den o. g. Informationen wurden 16 Vorhersagen mit jeweils unterschiedlichen Enddaten für das Training durchgeführt. Für jede Station wurden die maximalen und durchschnittlichen absoluten Fehler sowie die relativen durchschnittlichen Fehler gemessen.

  Wir haben 1000 Stationen mit Preisen ausgewählt und für jede Station ein Datum ausgewählt, aus dem 16 Mal mit unterschiedlichen Enddaten fürs Training vorhergesagt wurde. Für jede Station wurde der maximalen und durchschnittliche absolute Fehler sowie der relative durchschnittliche Fehler berechnet.

- Vorhersage ohne verfügbare Preise

  Wir haben 1000 Stationen mit Preisen ausgewählt und für jede Station einen Prädiktor mit einer alternativen Station vom Klassifier trainiert und 16 Mal Preise vorhergesagt. Die Preise der originalen Station wurden benutzt als Referenzwerte für die Berechnung der Fehler. Für jede Station wurde der maximale und der durchschnittliche absolute Fehler sowie der relative durchschnittliche Fehler berechnet.

- Vorhersagen ohne verfügbare Preise

  Wir haben 1000 Stationen mit verfügbaren Preisinformationen ausgewählt und für jede dieser Stationen einen Prädiktor mit einer alternativen Station vom Klassifier ausgesucht. Mit den o. g. Informationen wurden 16 Vorhersagen durchgeführt. Diesbezüglich wurden die Preise der originalen Stationen als Bezugswert für die Berechnung der Fehler benutzt. Für jede Station wurden die maximalen und durchschnittlichen absoluten Fehler sowie die relativen durchschnittlichen Fehler gemessen.

Der Benchmark wurde mit folgender Anweisung ausgeführt:

```
python benzlim benchmark  --nb_stations 1000 --nb_predictions 16
```

Durch die Ausführung der o. g. Anweisung werden die zwei Dateien `benchmark_with_prices.csv` und `benchmark_without_prices.csv` in `benzlim\out\` gespeichert.

Ein Abschnit aus den Ergebnissen ist in folgenden Tabellen gelistet, wo *e* der Unterschied zwischen einen prädizierten Preis p_p und den Referenzpreis p_r ist. Ein Abschnitt aus den Ergebnissen ist in der folgenden Tabelle aufgelistet. Hier ist "e" die Differenz zwischen der vorhergesagten Preisen "p_p" und der Referenzpreis "p_r".

- **Vorhersagen mit verfügbaren Preisen**

station_id	**6421**	**14554**	**6799**	**5049**	**10823**	**79**	**3607**	**12682**	**2885**
max(abs(e))	62	26	42	42	69	98	39	29	27
avg(abs(e))	30	18	29	15	34	27	24	20	20
avg(abs(e)/p $_r$)	0.023	0.0135	0.0202	0.0108	0.0237	0.0217	0.0191	0.0151	0.0161

Tabelle 1.

- **Vorhersagen ohne verfügbare Preise**

base_station_id	**2953**	**7655**	**58**	**30**	**0.0209**	**14018**	**15133**	**71**	**33**
used_station_id	7655	15133	13424	4	15164	14459	14184	14716	14184
max(abs(e))	58	71	46	72	60	39	43	37	45
avg(abs(e))	30	33	19	40	53	12	16	16	24
avg(abs(e)/p$_r$)	0.0209	0.0238	0.0144	0.0303	0.0325	0.0082	0.012	0.0124	0.0178

Im Durchschnitt haben Preisvorhersagen für sowohl Stationen mit Preisinformationen als auch Stationen ohne Preisinformationen eine absolute Fehlerrate von 25 bis 40.

Im Durchschnitt hatten Vorhersagen mit und ohne Preise einen absoluten Fehler von **33 +/- 6** und einen relativen Fehler von **0.022 +/- 0,004** . Absolute Fehler der Vorhersagen ohne Preise bewegen sich im selben Intervall.

Bekannte Probleme

- Der Speicherverbrauch ist proportional zur Anzahl der Prozessorkerne und kann beim Benchmarking mit sehr große Zahl an Stationen/Präditionen zu Problemen führen.
- Die Tankstrategie ist bei unrealistisch kleinen Tankkapazitäten teilweise fehlerhaft.
- "Multiprocessing" führt unter Windows zu Fehlern. Dementsprechend wird für Windows nur "Monoprocessing" verwendet.

Abschluss

Ausblick

- Die Anwendung von einem erweitertem Weg zur Berechnung der Unsicherheit könnte die Ergebnisse verbessern. Er könnte zum Beispiel auf den Prädiktionsfehler von für das Träning benutzte Preise

basiert sein.
- Die Tankstrategie benutzt zurzeit eine fixe Unsicherheit für alle Station/Anhaltspunkte. Es ist zu erwarten, dass sie mit genaueren Informationen bessere Schätzungen macht, nämlich die Unsicherheit jedes einzeln prädizierten Preis.
- Benzlim ist der Stützpunkt für viele weitere Projekte die ein effizienteres Routing für Autofahrer erbringen können. Diese wären bessere Routingalgorithmen, Reiseplanung Software usw.

Danksagung

Unser Dankwort geht an das Computational Healt Informatics (CHI), für das Hosting der InformatiCup an der Leibniz Universität Hannover.