Random Forest basierte Regression als mögliche Methode zur Parameterschätzung von Absatz-Zeit-modellen

-

Belegarbeit vorgelegt von

Faical Ghazali, Mat.-Nr.:4070433

Philipp Nostitz, Mat.-Nr.:4070743

Alexander Prinz, Mat.-Nr.:4069949

Hochschule Anhalt

Anhalt University of Applied Sciences

Zusammenfassung

Die Arbeit beinhaltet die Untersuchung des Random-Forest-Algorithmus als mögliche Methode zum Abschätzen eines Modellparameters zur Modellierung von Absatzratenveränderung bestimmter Artikel auf Basis vorliegender Verkaufsdaten. Das Modell basiert auf eine Funktion, welche das auch Abklingen diverser natürlicher Prozesse beschreibt, da das Verhalten der Absatzrate in bestimmten Zeitbereichen nachweislich ein abklingendes Schema aufweist. Zum Trainieren des Random Forest wurden Trainingsdaten mittels konventioneller Methoden zum Abschätzen von Modellparametern generiert und genutzt. Es konnte gezeigt werden, dass die angenommene Funktion zum Modellieren des Absatzrateverhaltens als sehr gut geeignet gelten kann. Die Ergebnisse hinsichtlich der Modellgüte des Random Forests waren sehr heterogen, wobei jedoch auch gute Modellschätzungen erreicht wurden konnten.

Abbildungsverzeichnis

Inhalt

Einleitung

Der Online-Handel ist eine in der heutigen Zeit nicht mehr wegzudenkende Möglichkeit bequem und unkompliziert Produkte oder Dienstleistungen via Internet zu erwerben (Stichwort E-Commerce). Der Online Markt wächst und es liegt somit klar auf der Hand, dass der Konkurrenzdruck somit auch im Online-Geschäft unter den Wettbewerbern ein großes Thema ist.

Das Data Mining hat sich in den letzten Jahren einen starken Stellenwert im Bereich des E-Commerce erarbeitet. Die Online-Händler versprechen sich durch die vielseitigen Methoden des Data Minings Wettbewerbsvorteile. Ziel des Data Minings ist das Generieren von weiteren nutzbringenden Daten aus den bereits vorhandenen Daten, um auf Basis dieser neu gewonnen Daten Optimierungen in Prozessen vornehmen zu können. Preise optimal profitabel anzupassen oder zielorientierte individuelle Werbemaßnahmen angepasst an Kundenvorlieben vorzunehmen.

Auf Basis der uns vorliegenden Verkaufsdaten aus dem Data Mining Cup Projekt wurde ein Konzept zum Modellieren von kurzfristig intensivierten Absatzverhalten entwickelt. Diese intensivierten Absätze zeichnen sich durch eine große Absatzrate am Starttag dieses Events aus, welche mit den Folgetagen abnimmt/abklingt. Die Basis des Konzeptes bildet das Modellieren dieses Abklingverhaltens in Anlehnung an das Abklingverhalten diverser in der Natur auftretender Prozesse. Das Modell wird durch die allgemeine Exponentialfunktion beschrieben, deren Parameter die Charakteristik des Modells bestimmen und bestimmt werden müssen.

Zur Bestimmung von Parametern werden konventionell Methoden aus dem Bereich des inversen Modellierens angewendet. Diese basieren auf Modellregression von bekannten Vorwärtsmodellen um implizit durch das Invertieren des Modells durch die bekannten Variablen (Ein- und Ausgabewerte) auf die Modellparameter abzubilden.

Die vorliegende Arbeit sieht vor, den konventionellen Ansatz durch einen alternativen Ansatz auf Basis eines Random-Forest-Algorithmus zu ergänzen bzw. auf den möglichen Nutzen dieser alternativen Methode zu untersuchen.

Da Random Forests als Teilgebiet des überwachten maschinellen Lernens Trainingsdaten benötigen und diese nicht explizit vorliegen, müssen diese in vorhergehenden Prozessschritten und alternativer Verfahren erzeugt werden. Dazu wird ein Vorwärtsmodell als Teilprozess eines größtenteils automatisierten Workflows in ein PYTHON-Programm implementiert. Durch eine Fitting-Funktion auf Basis des Vorwärtsmodells und der Gradientenabstiegsmethode wird aus vorprozessierten Daten in Form von Zeitreihenausschnitten aus Artikelverkaufsdaten bestimmter Artikel, der charakteristische Modellparameter bestimmt. Der Zeitreihenausschnitt sowie der Modellparameter bilden die Trainingsdaten für den Random Forrest. Ziel ist den Random-Forest-Algorithmus anhand der Trainingsdaten zu trainieren um auf weitere unabhängige daten diesen Parameter schätzen zu können.

Regressionsanalyse – eine kurze Zusammenfassung

Die Regressionsanalyse ist ein Verfahren aus dem Bereich der Statistik, das einen Zusammenhang zwischen den vorhanden Variablen untersucht. Dabei wird angenommen, dass eine abhängige Variable Y eine Funktion von unabhängigen Variablen X_1, X_2, usw. ist.

Dieser Zusammenhang stellt sich wie folgt dar:

$$Y = f(X_1, X_2, X_3, \dots, X_n)$$

Im einfachsten Fall basiert die Analyse auf einer Menge restriktiver Annahmen. Dazu gehören ein metrischen Skalenniveau, ein linearer Zusammenhang und das Vorliegen von Querschnittsdaten. Im einfachsten Fall kann zwischen den Variablen ein linearer Zusammenhang hergestellt werden. Man spricht hierbei von einer Linearen Regression. Diese Art der Regression kann erweitert werden. Das kann dazu führen, dass auch nominale oder ordinale Variablen untersucht werden können. Des Weiteren existiert eine Menge an Weiterentwicklungen. Diese ermöglichen zum Beispiel nominale dichotome abhängige Variablen oder ordinale abhängige Variablen zu untersuchen. Im ersten Fall spricht man dann von der Logistischen Regression, im Zweiten von der Ordinalen Regression.

Wird eine Analyse mit einer unabhängigen Variablen handelt es sich um eine einfache Regression, bei mehreren Variablen spricht man von einer multiplen Regression.

Neben der Herstellung eines Zusammenhangs zwischen abhängigen und unabhängigen Variablen, ist die Regressionsanalyse verwendbar, um Prognosen zu erstellen (Stoetzer, 2017).

Datengrundlage und Vorbereitung

Tabelle über alle Artikelverkaufszeitreihen

Als Datengrundlage dient ein Verzeichnis mit CSV-Dateien welche die jeweilige Verkaufszeitreihe eines Bestimmten Artikels umfasst. D.h. jeder Artikel hat eine eigene CSV-Datei. Diese enthält spalten und Zeilen wobei die erste Spalte den Zeitraum abbildet. Dieser Zeitraum umfasst konkret ein halbes Jahr und ist in täglichen Zeitabständen definiert wobei jede Zeile der Spalte also einen Tag entspricht. Die zweite Spalte der CSV-Tabelle entspricht den Verkaufszahlen des Artikels abgebildet auf den Verkaufstag. Also jede Zeile bildet wieder einen Tag ab. Somit ist sowohl das Verkaufsdatum als auch ein damit zusammenhängender Artikelverkauf bekannt. Das Prinzip entspricht einer Zeitreihe

Vorselektieren geeigneter Zeitreihen

Da der Datenbestand mehrere Tausend Artikel umfasst, jedoch viele Artikel über den gesamten Zeitraum nur sporadisch verkauft werden, wird vorab nach geeigneten Zeitreihen selektiert. Dazu dient ein PYTHON-Programm, welches über alle Artikelverkaufszeitreihen iteriert und nach Artikelzeitreihen sucht, welche eine festgelegte Mindestanzahl an insgesamt verkauften Artikeln aufweisen.

Die beiden folgenden Grafiken deuten die Unterschiede zwischen einzelnen Artikelverkaufszeitreihen an. Es sind deutliche Unterschiede in den Verkaufsraten also der Menge Verkaufter Artikel pro Tag zu erkennen. Zudem ist zu sehen, dass in beiden Beispielen die Verkäufe innerhalb einer bestimmten Zeit abspielen. Dabei ist anzunehmen, dass es sich um Werbemaßnahmen induzierte Verkäufe handelt.

Jede selektierte Tabelle wird abschließend an eine Gesamttabelle geheftet. Diese dient der weiteren Datenverarbeitung.

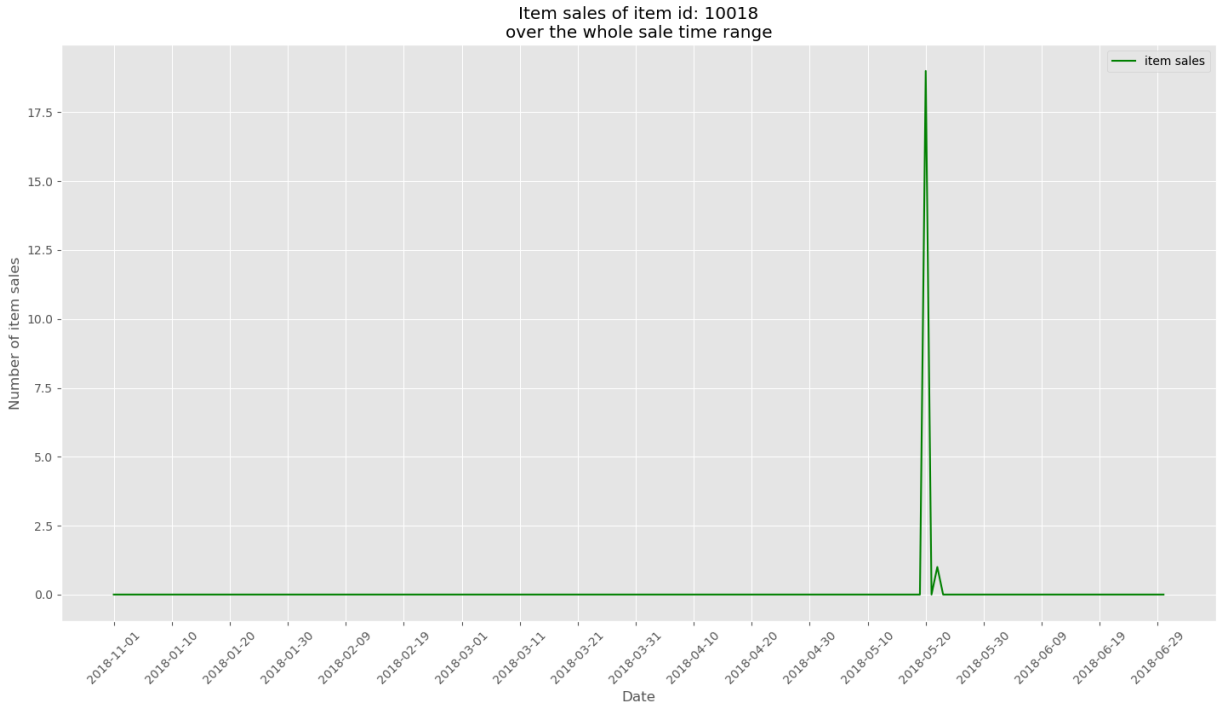


Abb. 1: Artikelverkaufszeitreihe zu Artikel mit der ID 10018

Dieser Artikel zeigt über den Gesamtzeitraum von einem halben Jahr nur sehr wenige Verkaufe

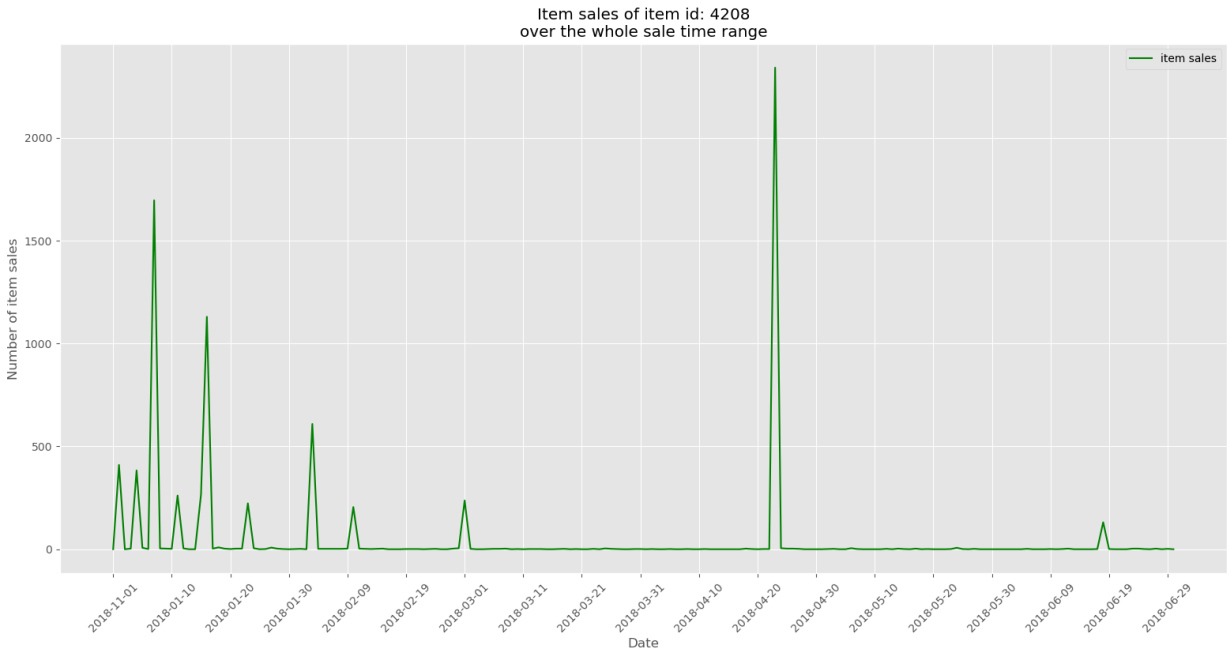


Abb. 2: Artikelverkaufszeitreihe zu Artikel mit der ID 4208

Dieser Artikel zeigt über den Gesamtzeitraum von einem halben Jahr verglichen mit den von Abbildung 1 weitaus mehr verkaufte Artikel. Diese zeigen sich vorrangig durch Verkaufs-Cluster. Also kurze Zeiträume in denen höhere Stückzahlen verkauft wurden.

Verkaufs-Cluster-Selektion

Nachdem die relevanten Verkaufszeitreihen vorselektiert wurden, werden die Verkaufs-Cluster lokalisiert. Dazu dient ein weiteres PYTHON-Programm, welches über die Tabelle der Vorselektierten Verkaufszeitreihen iteriert und mögliche Verkaufs-Cluster lokalisiert. Die Definition eines solchen Verkaufs-Cluster ist folgend. Ein Verkaufs-Cluster ist eine Teilverkaufszeitreihe bestehend aus Folgetagen und deren Verkaufszahlen. Also ein Ausschnitt der Gesamtverkaufszeitreihe wobei folgende Kriterien erfüllt sein müssen. Es müssen mindestens 3 Folgeglieder enthalten sein und jedes Folgeglied muss kleiner sein als das Vorhergehende. Zudem darf es nicht 0 sein.

Die folgende Abbildung (Abb. 3) zeigt einen Ausschnitt aus einer gesamten Verkaufszeitreihe und 2 darin lokalisierbare Verkaufst-Cluster.

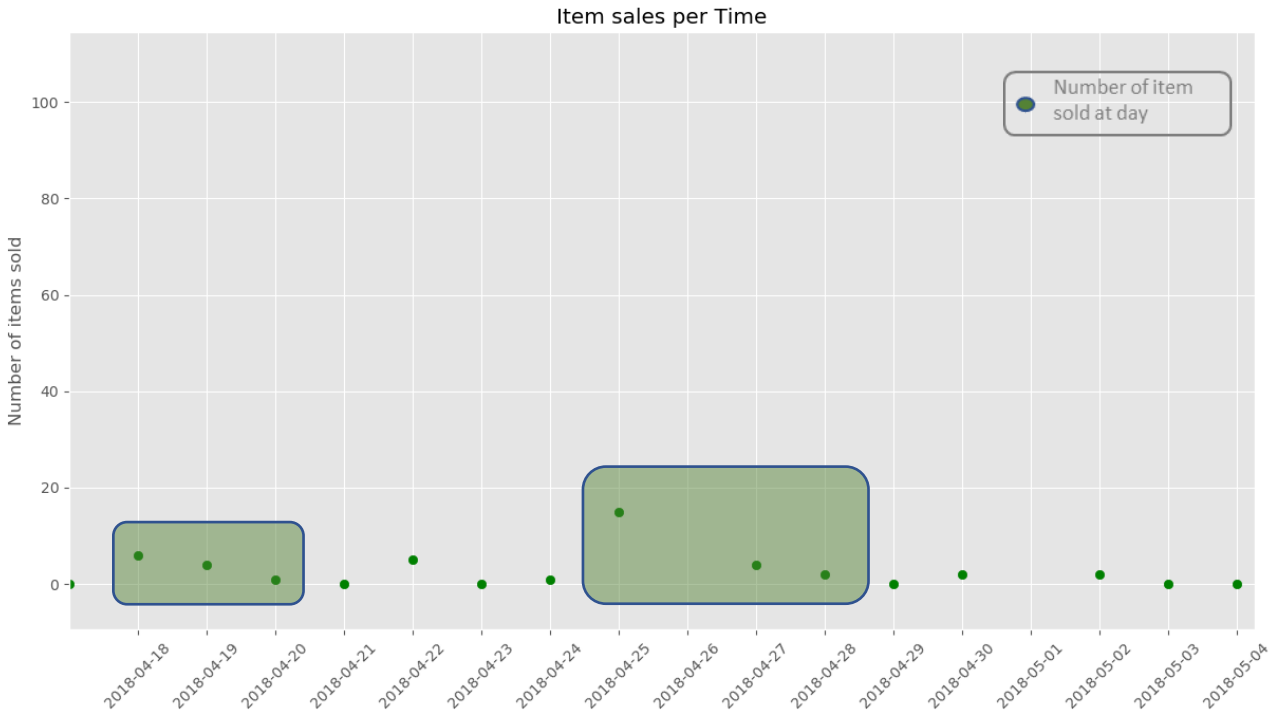


Abb. 3: Zeitausschnitt einer Verkaufszeitreihe mit 2 lokalisierbaren Verkaufs-Cluster

Beide Cluster weisen die gewünschten Kriterien auf. Sie bestehen beide aus 3 Elementen und die Folgeglieder weisen eine geringere Zahl auf als die vorhergehenden. Dies ist wichtig bzgl. des angenommen Abklingverhaltens der Verkaufsrate.

Das Verkaufsabklingverhalten-Modell – Die Natur als Motivator

Ein Großteil von auf mehrere Tage verteilte Artikelverkäufe mit signifikant hohen Verkaufszahlen pro Tag weisen ein absteigendes Verhalten seitens der Verkaufszahlen für die Folgetage auf. D.h., dass es einen Starttag gibt ab welchen die Artikel in bedeutend höherer Menge abgesetzt werden als üblich und in den Folgetagen nimmt mit jedem Tag die Verkaufszahl ab. Es scheint also, als würde die Verkaufsrate gedämpft werden. Geht man davon aus, dass diese Verkaufs-Cluster als Ereignis etwa auf Werbemaßnahmen antworten, so könnte man die Überlegung anstellen, dass die enthusiastischsten Käufer am ersten Tag kaufen, während in den Folgetagen die etwas trägeren Käufer kaufen.

Das Abklingen von instationären also zeitlich varianten Prozessen ist in der Natur sehr häufig anzutreffen. So klingt der radioaktive Zerfall mit der Zeit ab, weil die Anzahl an noch nicht zerfallenen Kernen langsam jedoch nicht linear abnimmt (Clauser, 2016).

Auch das Abklingen der Amplitude eines harmonisch schwingenden gedämpften Pendels klingt nichtlinear ab. Es gibt eine Reihe weiterer Prozesse, dessen Erwähnung jedoch nicht Bestand dieser Arbeit sein sollen. Abbildung 4 zeigt jedoch anschaulich den Amplitudenabfall eines harmonisch schwingenden gedämpften Pendels.

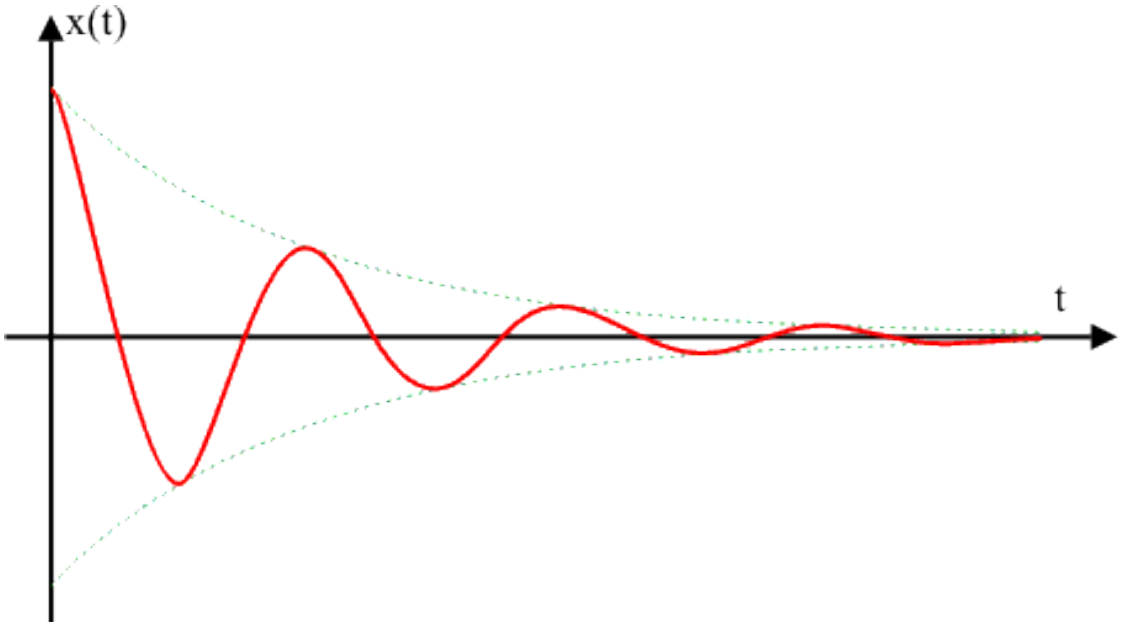


Abb. 4: Amplitudenverhalten eines harmonischen gedämpften Pendels mit der Zeit

Die rote Kurve zeigt die Amplitude des Pendels, welche mit der Zeit abnimmt. Die beiden grünen gepunkteten Kurven beschreiben jeweils eine Funktion, die die Extrema der Amplitude tangiert. Die Funktion der maximalen Amplituden ist mathematisch identisch mit der Funktion, welche den Radioaktiven Zerfall beschreibt.

Alle beschriebenen abklingenden Prozesse können hinsichtlich der Dämpfung ihrer Intensität durch folgende Funktion beschrieben werden.

$$N(t) = N_0 e^{-\lambda t} \tag{1}$$

Es handelt sich dabei um die allgemeine Exponentialfunktion mit negativen Exponenten. Die unabhängige Variable ist der Zeitpunkt t. Der Funktionswert ist die Intensität oder Anzahl des Prozesses. Die beiden Funktionsparameter sind N_0 der Anfangswert der Intensität zum Zeitpunkt t=0, sowie der Parameter λ, welcher die Dämpfungscharakteristik bestimmt. Die Dämpfung selbst ist also einerseits von dem Anfangswert, etwa im Beispiel dieser Arbeit von der Absatzrate am ersten Tag eines Verkaufs-Clusters abhängig, sowie aber auch vom Parameter λ, welcher im konkreten Fall unbekannt ist und geschätzt werden soll. Dieser Parameter charakterisiert das Verhalten der Absatzrate über die Zeit maßgeblich.

Generierung der Trainingsdaten

Die λ-Werte sollen pro Verkaufs-Cluster vom Random-Forest-Algorithmus geschätzt werden. Als Eingabewerte dienen die Verkaufs-Cluster. Jedoch müssen bei den Trainingsdaten auch die damit verbundenen λ-Werte bekannt sein. Dazu wird mithilfe eines selbstimplementierten PYTHON-Programms jeder λ-Wert pro Verkaufs-Cluster via Modellinvertierung auf Basis von Modellregression durch das Gradientenabstiegsverfahren geschätzt. Der Grund dafür keine bereits existierende Fitting-Funktion aus bekannten Bibliotheken zu benutzen war der, dass diese nicht flexibel genug bzgl. Parametereinschränkungen sind. Zwar sind 2 Parameter in der Abklingfunktion enthalten. Jedoch muss nur der λ-Wert geschätzt werden, da der erste Parameter der Anfangswert jedes Verkaufs-Clusters ist und somit bereits bekannt ist.

Die existierenden Fitting-Funktionen beinhalten aber grundsätzlich das Schätzen jedes Parameters. Die eigens implementierte Fitting-Funktion arbeitet sehr gut und wird mittels des Bestimmtheitsmaßes geprüft.

Im Folgenden sind einige Modellbeispiele zu sehen, welche größtenteils eine sehr gute Modellgüte gemessen am Bestimmtheitsmaß sowie dem Kurvenverlauf der Modellkurve aufweisen. Im Anhang sind alle Visualisierungen der Modellfälle zu finden.

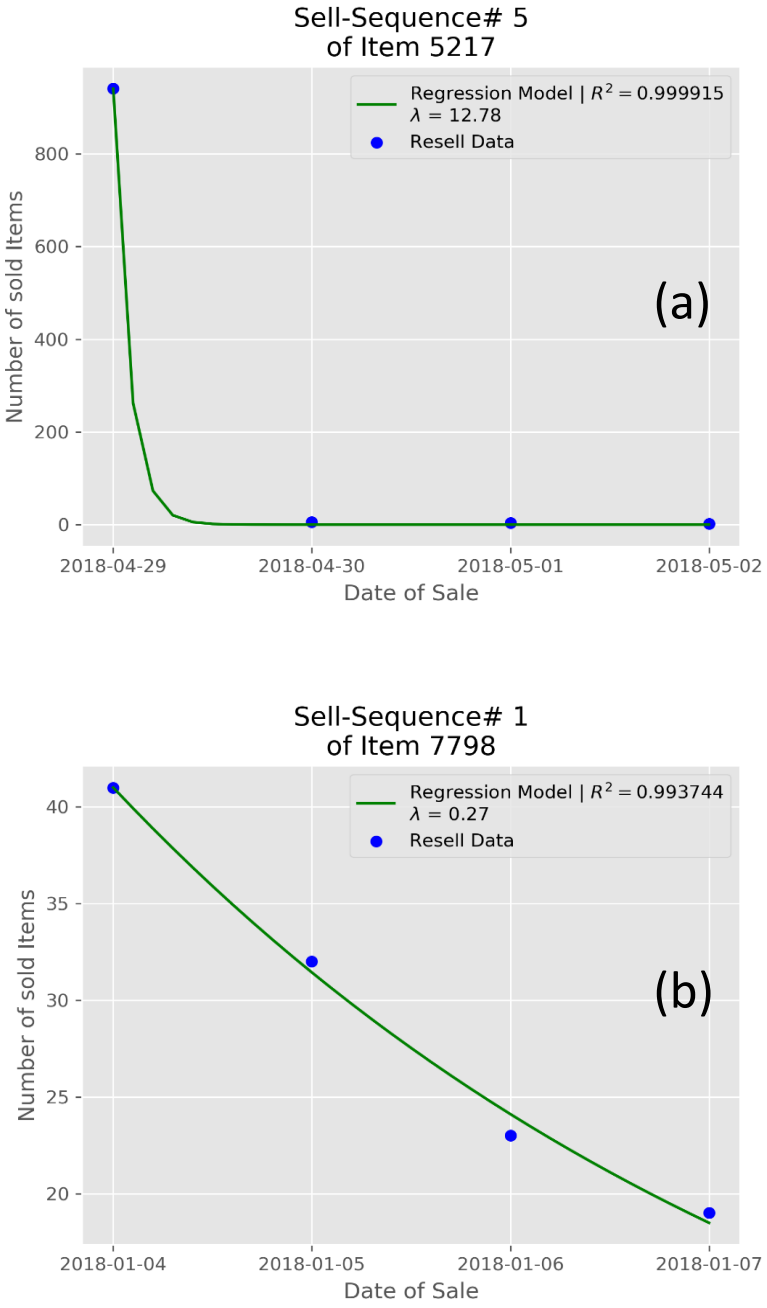


Abb. 5: Visualisierungen der Modellanpassungsergebnisse

Beide Grafiken zeigen durch die blauen Punkte die Daten des entsprechenden Verkaufs-Clusters. Sie bilden auf die Verkaufstage die jeweiligen Verkaufszahlen ab. Der grüne Graf zeigt jeweils das angepasste Modell. Für beide Modelle zeigt sich eine sehr gute Güte der Modellanpassung welche durch die entsprechende Zahl des Bestimmtheitsmaßes (siehe Legende) nochmals numerisch unterstrichen wird.

Es wurden insgesamt 80 Modelle erzeugt und deren λ-Werte geschätzt. Dabei konnten für über 75% der Modelle ein R^2 -Wert von zwischen 0.95 und 1 erreicht werden (siehe dazu Histogramm in Abbildung 6).

Demzufolge ist die Hypothese, dass das zeitliche Absatzratenverhalten durch die Abklingfunktion modelliert werden kann, als wahr zu bewerten. Die so erzeugten λ-Werte wurden zusammen mit den dazugehörenden Verkaufs-Clusterdaten als CSV-Dateien gespeichert, um sie dem eigentlichen Kernprozess zuführen zu können.

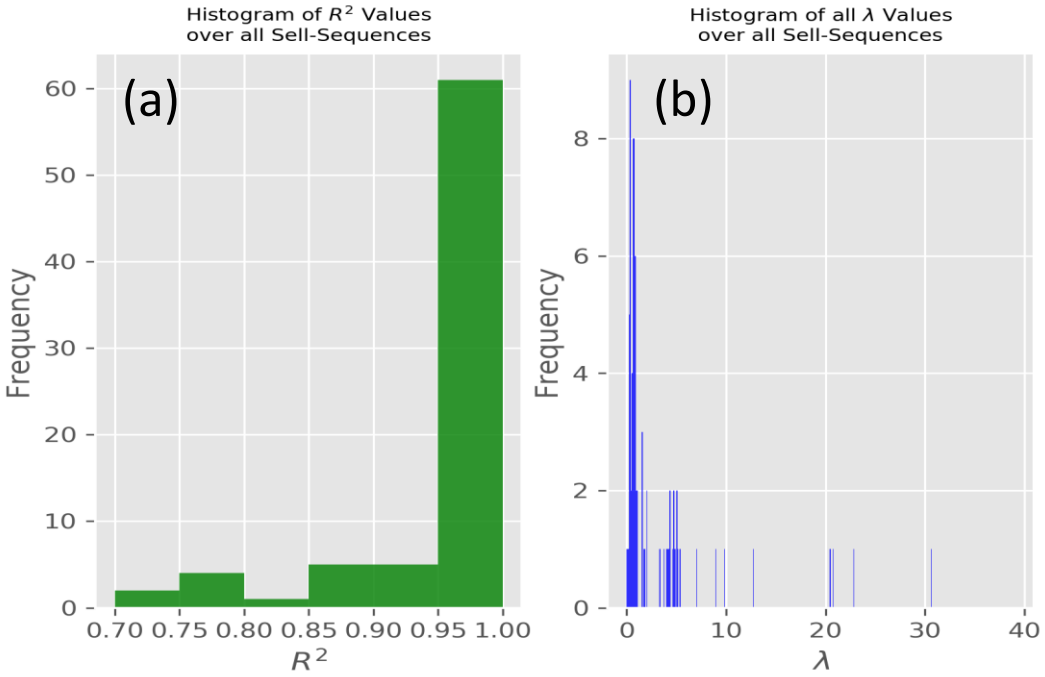


Abb. 6: Histogramme über alle R^2 und λ-Werte

Grafik a zeigt ein Histogramm über alle Bestimtheitsmaßwerte aller Modellfälle. Es zeigt sich, dass der überwiegende Teil in einem Bereich zwischen 0.95 und 1 liegt und somit als signifikant gut angesehen werden kann. Die Annahme, dass sich das Verkaufsabklingverhalten durch die Abkling-Funktion modellieren lässt, ist somit begründet. Das zweite Histogramm zeigt die Häufigkeit der λ-Werte. Hierbei überwiegen Werte im Bereich zwischen 0 und 1, sowie eine weitere Ansammlung zwischen 4 und 6.

Random Forest

Grundlagen

Der Random Forest ein Klassifikationsalgorithmus, der eine Erweiterung der Entscheidungsbäume darstellt. Im Gegensatz zu diesen besteht er nicht nur aus einem Baum, sondern aus einer Ansammlung unkorrelierter Bäume. Deswegen zählen sie zu der Klasse der Ensemble-Klassifikatoren. Random Forests eignen sich aber nicht nur für Klassifikationen, sie können auch in der Regressionsanalyse oder im unüberwachten Lernen eingesetzt werden (Liaw, 2001).

Entscheidungsbäume zählen zu den beliebtesten maschinellen Lernverfahren, da sie sowohl vom Menschen als auch von Computern, aufgrund ihrer Struktur, sehr gut interpretierbar sind. Um eine Entscheidung mithilfe eines Entscheidungsbaumes zu treffen, wird ein Pfad entlang von der Wurzel bis zu einem Blatt gegangen. An jedem Knoten wird eine Entscheidung entsprechen des zugewiesenen Attributes getroffen und der entsprechende weiterführende Pfad eingeschlagen.

In der Regel arbeitet der Algorithmus mit Bootstrap-Sampling. Die Art des Samplings berechnet wiederholt Statistiken, denen aber nur eine Stichprobe zugrunde liegt. Im einfachsten Fall werden die Bootstrap-Stichproben dadurch generiert, dass je Ziehung *n* Mal aus der gegebenen Stichprobe ein Wert mit zurücklegen gezogen wird. Die Verteilung der gesamten Stichprobe wird durch die Verteilung in der Bootstrap-Stichprobe approximiert (Efron, 1993).

Der Random Forest Algorithmus arbeitet wie folgt:

1. Generiere *n* Bootstrap-Samples
2. Für jede dieser Proben wird ein nicht-vereinfachter Regressions- oder Entscheidungsbaum erzeugt. Bei den üblichen Entscheidungsbäumen erfolgt die Aufteilung an Knoten anhand der besten Aufteilung über alle Variablen. Bei der Generierung eines Random Forest werden zufällig Variablen gewählt, zwischen denen die Aufteilung erfolgt.
3. Triff eine Aussage, indem die Aussagen der einzelnen Bäume aggregiert werden. Wird der Random Forest-Algorithmus für die Klassifizierung genutzt, entspricht die Aussage der Mehrheit der abgegebenen Aussagen. Bei einer Regression wird ein Mittelwert über alle Ergebnisse der Einzelbäume erstellt.

Die Strategie für das Erzeugen der Einzelbäume ist zwar kontraintuitiv, allerdings ist die Performance der Random Forest im Vergleich zu anderen Klassifikationsalgorithmen, wie z.B. Neuronale Netze und Support Vector Maschinen, besser. Darüber hinaus ist der Algorithmus robust gegenüber Over-fitting (Liaw, 2001).

Durchführung der RandomForest Regression

Die oben beschrieben Trainingsdaten sollen nun dem Random Forest Algorithmus zum Trainieren übergeben werden.

Die Umsetzung der Lernaufgabe erfolgt demnach in Python. Hierfür wurde die Sci-Kit-Learn Bibliothek verwendet, um den Algorithmus nicht selbst implementieren zu müssen.

Diese liest die generierten Trainingsdaten ein und fügt sie mittels eines Outer-Joins zusammen. Während des Einlesens wurde eine Spalte eingefügt, die Auseinanderhaltung der einzelnen Verkaufsphasen ermöglicht.

```
60  ###############################################################################
61  """
62  1.BILDEN DER TRAININGS UND TESTSETS
63  """
64  ###############################################################################
65
66
67  train_set = result[result['man_index'] >=10].copy()
68  test_set = result[result['man_index'] < 10].copy()
69  train_set_labels = train_set[['man_index','lambda']].copy()
70  test_set_labels = test_set[['man_index','lambda']].copy()
71  train_set_len = len(train_set['man_index'].unique())
72  test_set_len = test_set['man_index'].unique()
73
74  ###############################################################################
75  """
76  RANDOM FOREST 1.VERSUCH TRAINING
77  """
78  ###############################################################################
79
80  rand_forest = RandomForestRegressor(
81    bootstrap = True,
82    n_estimators = 100,
83  )
84
85  rand_forest.fit(train_set[['N items at day']], train_set_labels[['lambda']])
86
87
```

Abb. 7: Aufteilung in Trainings- und Testdaten

In dieser Abbildung ist die Bildung der Test- Und Trainingssets abgebildet. In der ersten Zeile werden alle Sequenzen mit dem Index größer gleich 10 den Trainingsdaten zugewiesen. Die Restlichen, in der nachfolgenden Zeile den Test-Daten.
Im unteren Teil ist zum einen die Initialisierung des RandomForestRegressors mit den entsprechen Parametern zu sehen. Zum anderen werden in der letzten Zeile die Trainingsdaten und die entsprechenden Labels übergeben.

Im nächsten Schritt werden die Daten in Trainings- und Testmengen unterteilt. Um in den Ergebnissen Heterogenität herzustellen wurden diese Aufteilung zweimal vorgenommen.

Im ersten Durchgang bildeten die hinteren 18 Sequenzen die Testmenge und im zweiten Durchlauf die vorderen 10.

Doch bevor der Algorithmus benutzt werden kann muss der RandomForestRegressor initialisiert werden. Die oben genannte Bibliothek bietet bei der Initialisierung des Regressors eine Vielzahl an Übergabeparametern. Neben der Möglichkeit Bottstrap-Sampling zuzulassen, kann auch die Anzahl der zu erstellen Bäume festgelegt werden. Des Weiteren besteht die Möglichkeit ein Kriterium für den bestmöglichen Split festzulegen oder welche Tiefe ein durch den Algorithmus generierter Baum maximal haben darf, um nur einige Beispiel zu nennen. Dabei wurde das oben beschriebene Bootstraping zu gelassen, die Anzahl der zu erstellenden Bäume wurde auf 100 gesetzt. Beides entspricht dem Default-Wert und wurde zu Anschauungszwecken eingetragen.

Nun können die Trainingsdaten entsprechen Zeile 85 Abb. 7 dem Regressor zum Lernen übergeben werden.

Im letzten Schritt soll nur der angelernte Regressor verwendet werden, um Prognosen treffen zu können. Abbildung 8 zeigt eine solche Abfrage exemplarisch. In Zeile 94 iteriert eine Schleife über die einzelnen Verkaufscluster. Es werden in Zeile 95 die Anzahlen der verkauften Artikel der jeweiligen Cluster übergeben. Zeile 96 und 97 zeigen eine optionale Ausgabe. Und im letzten Schritt wird der geschätzte λ-Wert in die Sequenz des DataFrames eingetragen.

```
88  ###############################################################################
89  """
90  RANDOM FOREST 1.VERSUCH VORHERSAGE
91  """
92  ###############################################################################
93
94  for j in test_set_len:
95    pred = rand_forest.predict(test_set[test_set['man_index']==j][['N items at day']])
96    #print(pred[0])
97    #print(test_set[test_set['man_index']==j][['N items at day', 'lambda']])
98    test_set.loc[test_set['man_index'] == j, 'rf_lambda'] = pred[0]
99
100
```

Abb. 8: Prognose der λ-Werte

Die Abbildung zeigt die Abfrage einer eines Lambda-Wertes. In Zeile 95 werden die Anzahlen der verkaufen Artikel eines Verkaufsclusters übergeben.
In Zeile 98 wird die generierte Schätzung in den DataFrame eingetragen.

Durchführung der RandomForest Regression

Die oben beschrieben Trainingsdaten sollen nun dem Random Forest Algorithmus zum Trainieren übergeben werden.

Dieser Schritt erfolgt in einem separaten Python-Skript

Dieses liest die generierten Trainingsdaten ein und fügt sie mittels eines Outer-Joins zusammen. Während des Einlesens wurde eine Spalte eingefügt, die Auseinanderhaltung der einzelnen Verkaufsphasen ermöglicht.

Im nächsten Schritt werden die Daten in Trainings- und Testmengen unterteilt. Um in den Ergebnissen Heterogenität herzustellen wurden diese Aufteilung zweimal vorgenommen.

Im ersten Durchgang bildeten die hinteren 18 Sequenzen die Testmenge und im zweiten Durchlauf die vorderen 10.

Doch bevor der Algorithmus benutzt werden kann muss der RandomForestRegressor initialisiert werden. Dabei wurde das oben beschriebene Bootstraping zu gelassen, die Anzahl der zu erstellenden Bäume wurde auf 100 gesetzt, was dem Default-Wert entspricht.

Die über den Regressor geschätzten Lambda-Werte wurden dem DataFrame hinzugefügt.

Ergebnisse

Insgesamt wurden 18 modellunabhängige Verkaufs-Cluster getestet und auf ihre Modellgüte untersucht.

Abbildung 9 zeigt exemplarisch 4 Beispiele der Modellergebnisse, bei denen das Random-Forest-Modell sowohl gegen das Referenzmodell, welches zum Generieren der Trainingsdaten verwendet wurde, als auch gegen die Verkaufsdaten verglichen wird.

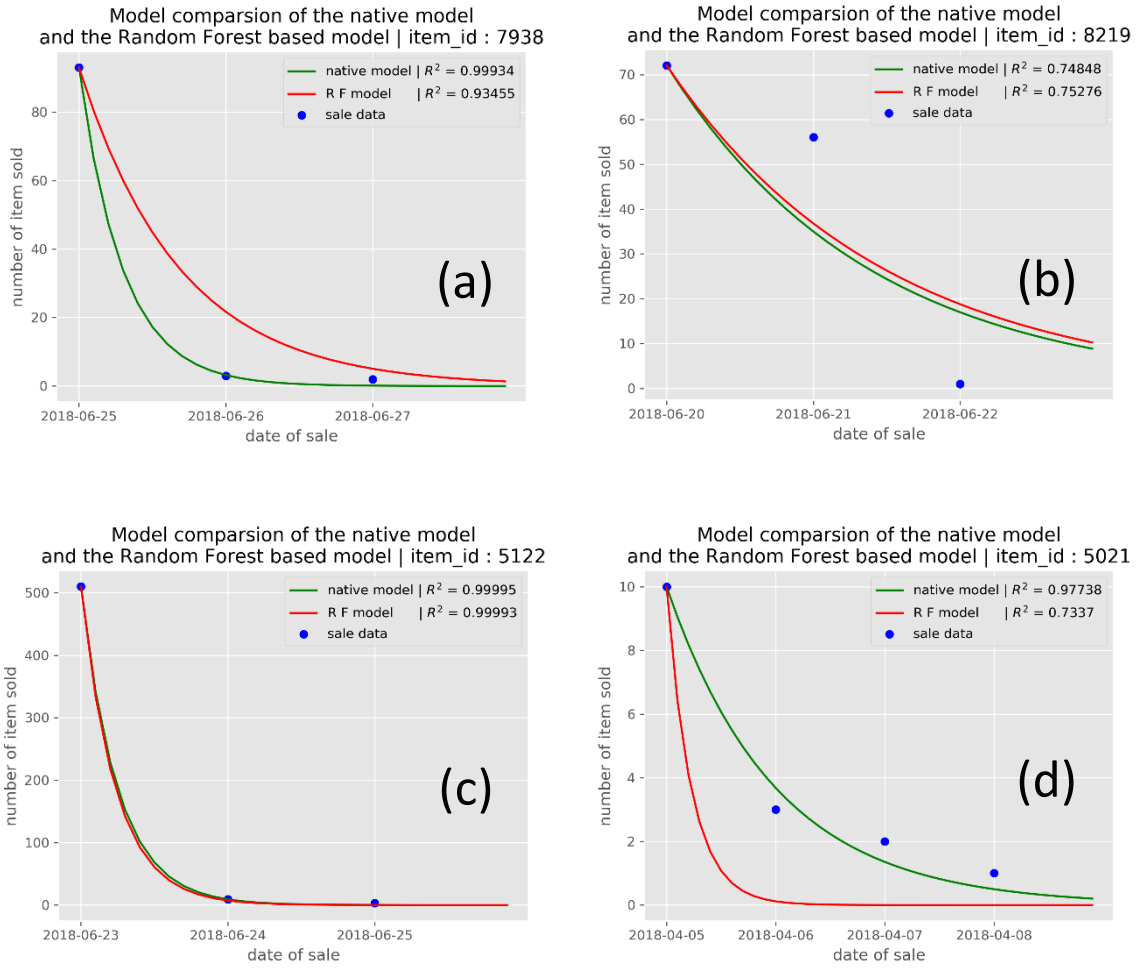


Abb. 9: Exemplarische Visualisierungen der Ergebnisse.

Die 4 Visualisierungen zeigen als blaue Punkte die Verkaufsdaten eines Verkaufs-Clusters. Die grüne Kurve repräsentiert das konventionelle Model, mit welchem auch die Trainingsdaten generiert wurden. Die Rote Kurve beschreibt das Random Forest Modell. Beispiel b und c zeigen eine gute Modellgüte, da die Kurven sehr gut die Verkaufsdaten approximieren. Die beiden Modell a und d zeigen jedoch eine schlechtere Modellgüte. D.h. die für diese Modelle geschätzten λ-Werte sind also weniger gut geschätzt wurden. Insgesamt lässt sich die jeweilige Modellgüte auch über den Vergleich beider R^2-Werte ablesen (siehe Legende).

Als Referenzmodell dient dabei das Modell, mit welchem die Trainingsdaten erzeugt wurden.

Die Modellgüte des Random-Forest-Modells ist dabei recht heterogen. D.h. es gibt Modelle mit sehr guter Modellgüte, aber auch Modelle mit geringerer Modellgüte.

Um einen besseren Vergleich der Modelle ziehen zu können, wurden für jeden Verkaufs-Fall die R^2 - Werte für jedes Modell ermittelt und die Gesamtmenge der R^2 Werte in Form eines Boxplots statistisch untersucht, wobei auch die R^2-Werte des Referenzmodells untersucht wurden.

Der Medianwert des Random-Forest-Modells liegt mit 0.8946 in einem guten Bereich. Jedoch zeigt sich eine hohe Varianz der R^2-Werte. Modellwerte, welche eher nicht gut sind, weichen dann besonders stark vom Vergleichswert ab.

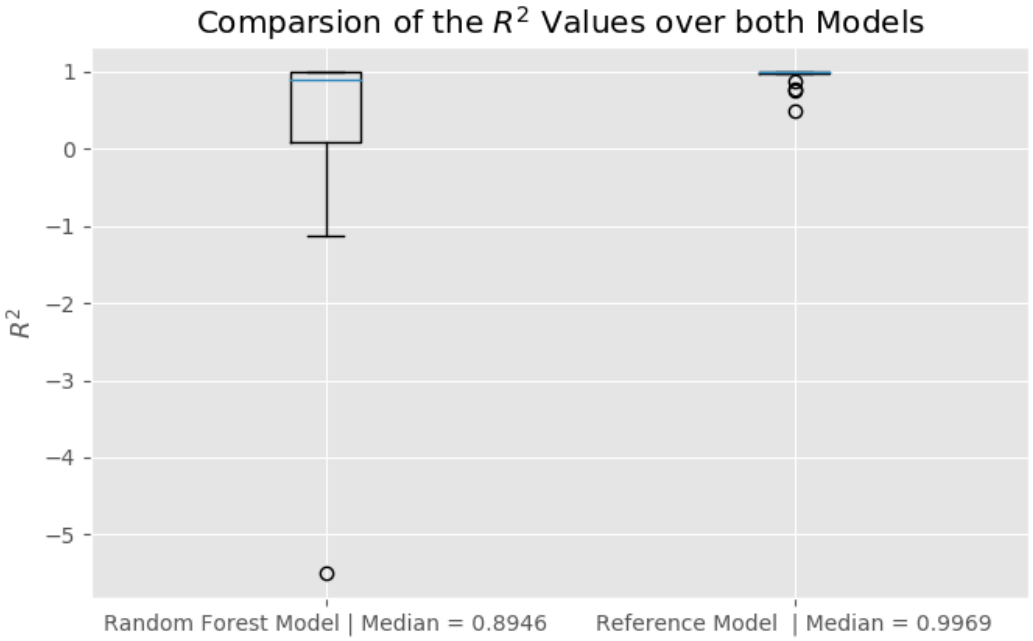


Abb. 10: Boxplot zum Visualisieren der Modellgüte.

Der Plot besteht aus zwei Boxplot-Elementen. Das erste Element visualisiert das Häufigkeitsverhalten der R^2-Werte bzgl. des Random-Forest-Models. Das zweite Element visualisiert das Häufigkeitsverhalten des Referenzmodells. Das Random-Forest-Modell weist gegenüber dem Referenzmodell eine wesentlich stärkere Varianz der R^2-Werte auf. Dennoch liegt der Median in einem guten Bereich.

Ausblick

Die Ergebnisse bzgl. der Modellgüte sind recht heterogen. D.h. es sind sowohl gute als aber auch weniger gute Modellparameterschätzungen zu finden.

Der Grund könnte darin liegen, dass insgesamt zu wenig Trainingsdaten zur Verfügung standen und die Trainingsdaten bzgl. ihrer Ähnlichkeit sehr divers waren. Das zeigt sich auch durch die in Abbildung 6 b visualisierten λ-Werte. Diese häufen sich in einem Bereich zwischen 0 und 2, sowie in einer weiteren Ansammlung zwischen 3 und 6. Jedoch gibt es auch vereinzelte Werte in höheren Bereichen.

Möglicherweise kann das Random-Forest-Model aufgrund zu weniger Trainingsdaten nicht gut genug die unterschiedlichen Modelle trainieren. Diese Unterschiede können mit der zum einen heterogenen Verteilung, teilweise jedoch auch schematisch anmutenden Verteilung der λ-Werte zusammenhängen.

Es kann zusammenfassend gesagt werden, dass der Random-Forest-Algorithmus ein Potenzial als Methode zum Schätzen von Parametern hat. Dies zeigen Beispiele mit guter Modellgüte. Jedoch sind größere Mengen von Trainingsdaten nötig um dieses Potenzial auszuschöpfen.

Literaturverzeichnis

CLAUSER, C., (2014), „Einführung in die Geophysik“, Springer Spektrum, DOI: 10.1007/978-3-662-46884-5

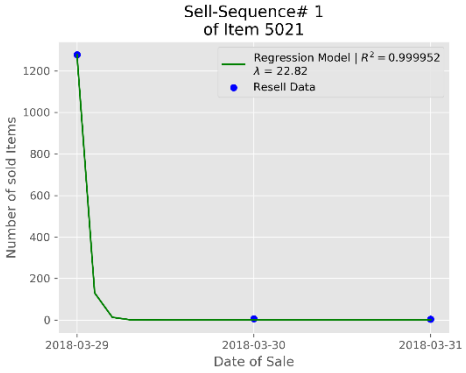
LIAW, A., WIENER, W., (2002), „Classification and Regression by randomForest“, R News Volume 2/3, December 2002, ISSN: 1609-3631

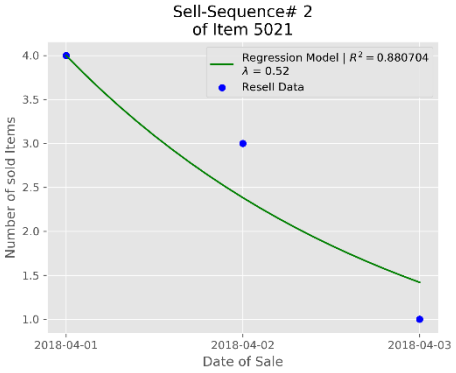
STOETZER, W.-M. (2017), „Regressionsanalyse in der empirischen Wirtschafts- und Sozialforschung Band 1“, Springer Gabler, DOI: 10.1007/978-3-662-53824-1

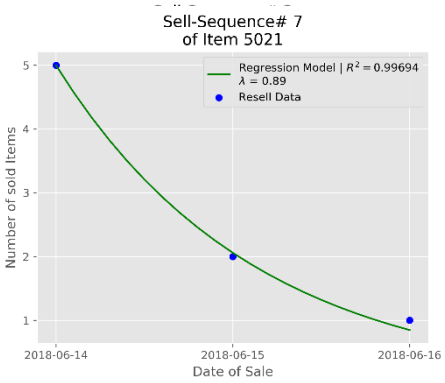
BREIMAN, L. (2001), „Random Forests“, Kluwer Academic Publishers, DOI: 10.1023/A:1010933404324

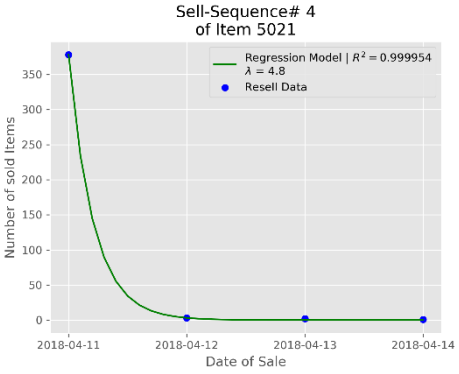
Anhang

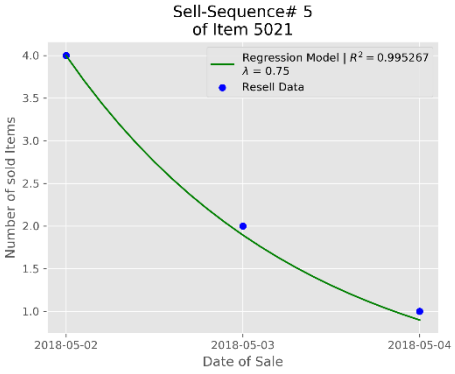
Ergebnisse der Parameterschätzung | Item_ID 5122

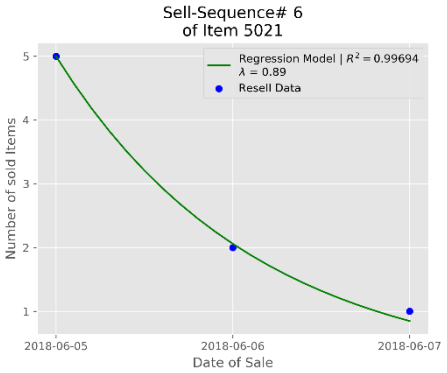




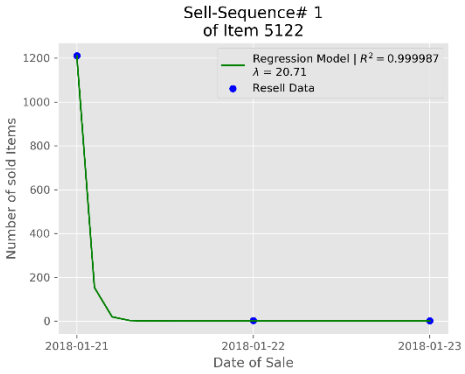


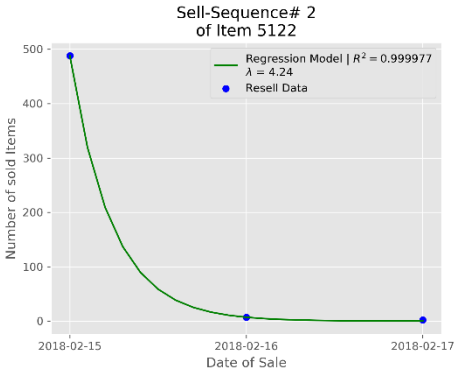


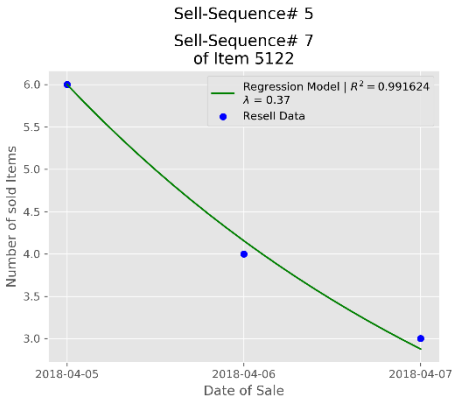


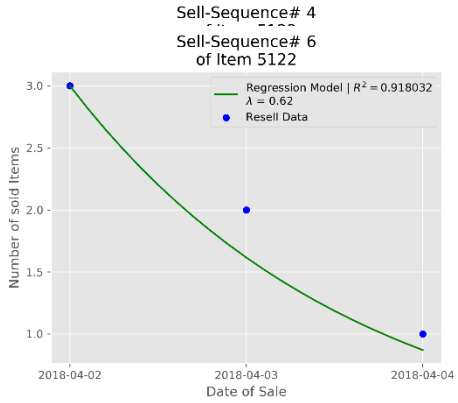


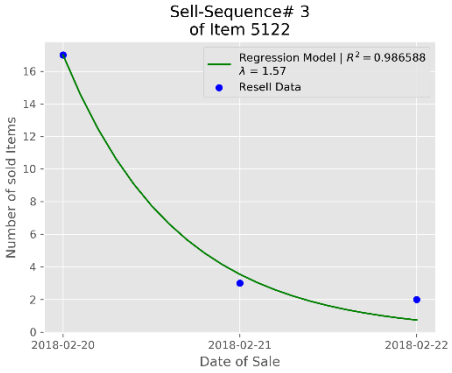
Ergebnisse der Parameterschätzung | Item_ID 5021

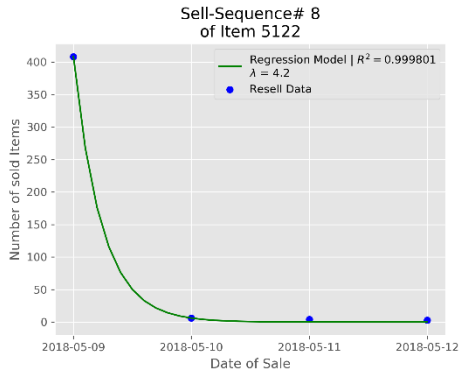


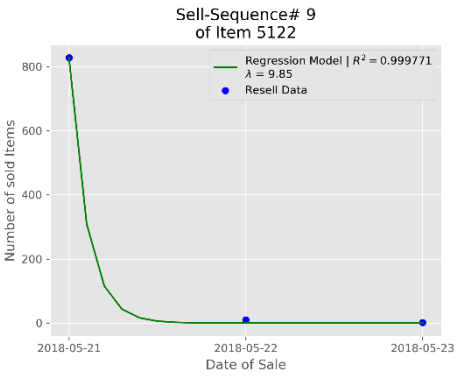


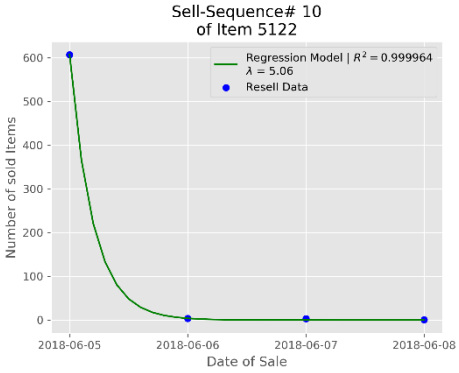


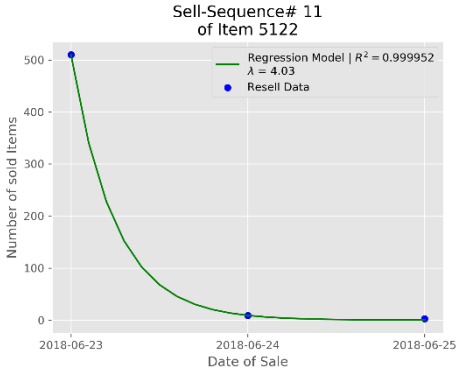




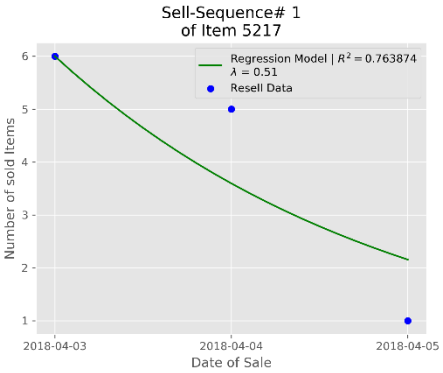


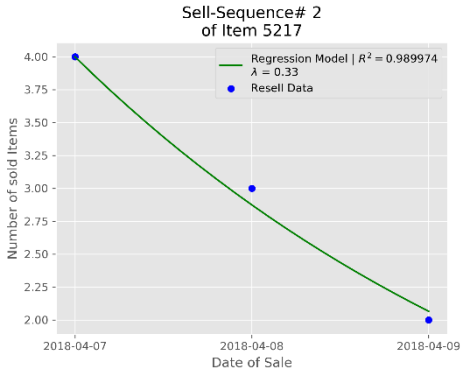


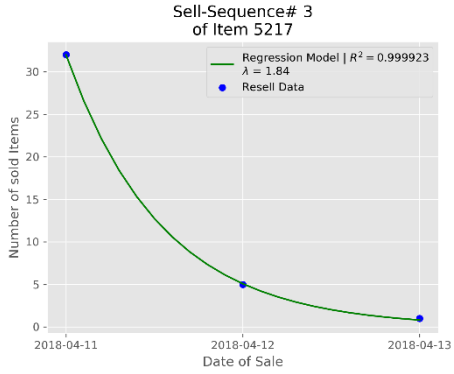


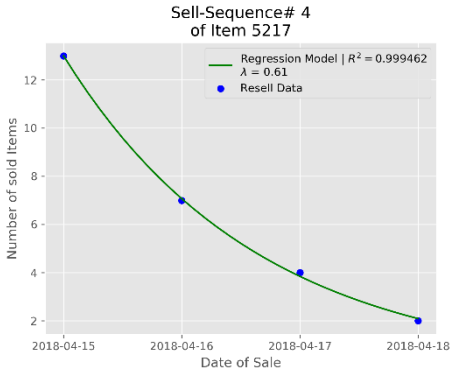


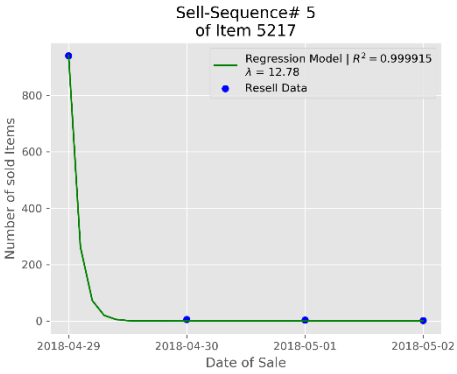
Ergebnisse der Parameterschätzung | Item_ID 5217

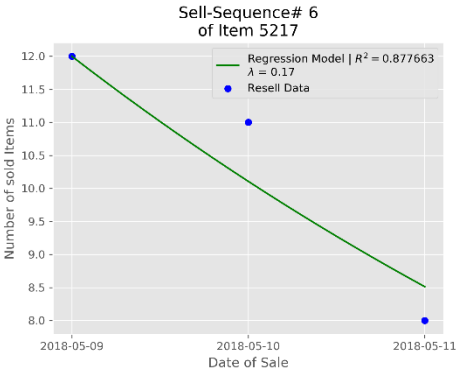


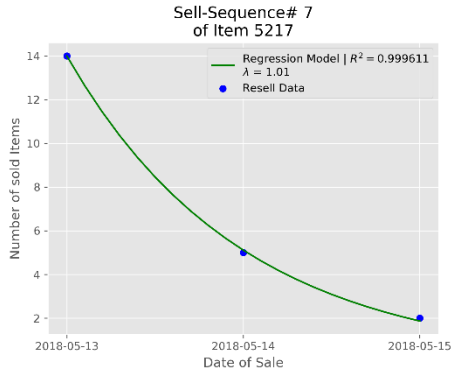


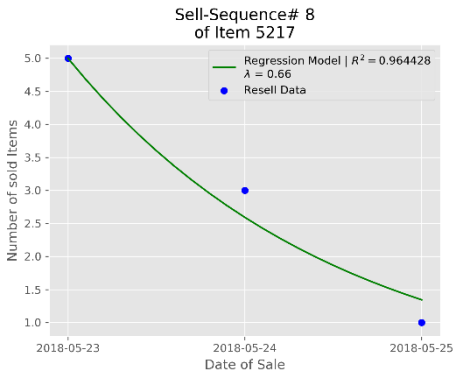

Sell-Sequence# 3
of Item 5217
Regression Model | $R^2 = 0.999923$
$\lambda = 1.84$
Resell Data
Number of sold Items
Date of Sale
2018-04-11
2018-04-12
2018-04-13


Sell-Sequence# 4
of Item 5217
Regression Model | $R^2 = 0.999462$
$\lambda = 0.61$
Resell Data
Number of sold Items
Date of Sale
2018-04-15
2018-04-16
2018-04-17
2018-04-18

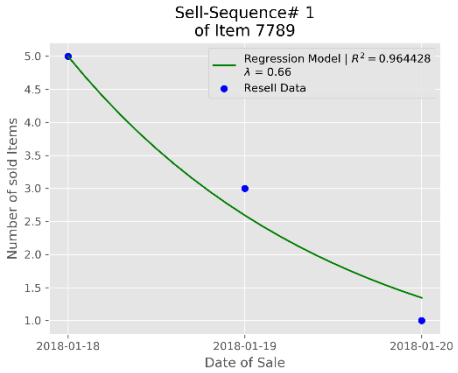

Sell-Sequence# 5
of Item 5217
Regression Model | $R^2 = 0.999915$
$\lambda = 12.78$
Resell Data
Number of sold Items
Date of Sale
2018-04-29
2018-04-30
2018-05-01
2018-05-02

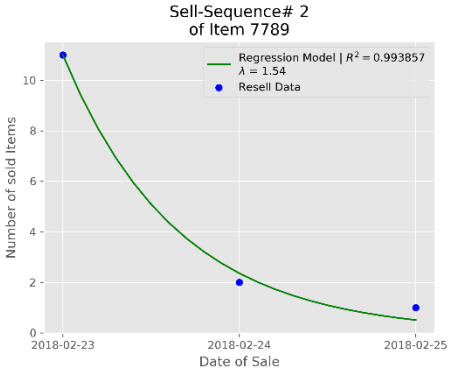

Sell-Sequence# 6
of Item 5217
Regression Model | $R^2 = 0.877663$
$\lambda = 0.17$
Resell Data
Number of sold Items
Date of Sale
2018-05-09
2018-05-10
2018-05-11

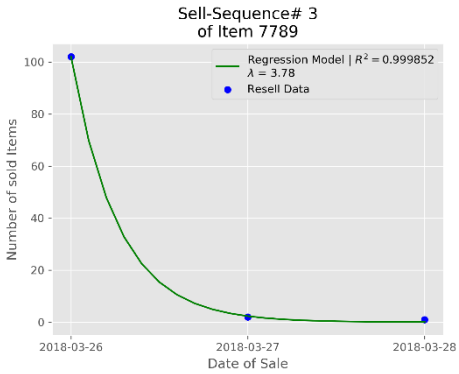

Sell-Sequence# 7
of Item 5217
Regression Model | $R^2 = 0.999611$
$\lambda = 1.01$
Resell Data
Number of sold Items
Date of Sale
2018-05-13
2018-05-14
2018-05-15

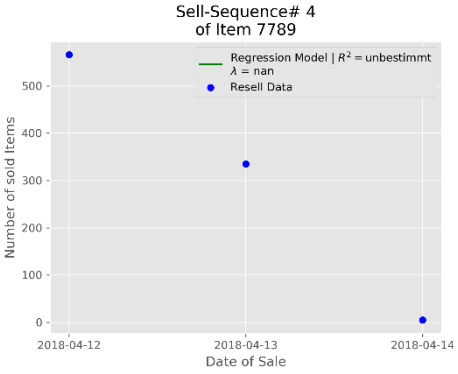

Sell-Sequence# 8
of Item 5217
Regression Model | $R^2 = 0.964428$
$\lambda = 0.66$
Resell Data
Number of sold Items
Date of Sale
2018-05-23
2018-05-24
2018-05-25

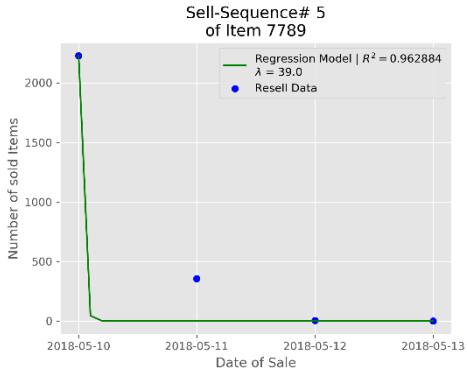
Ergebnisse der Parameterschätzung | Item_ID 7789

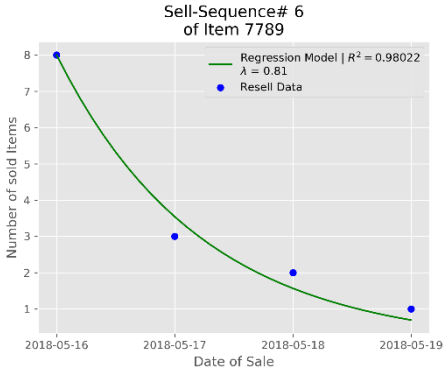


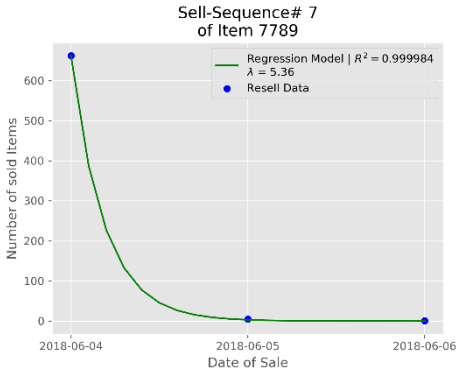


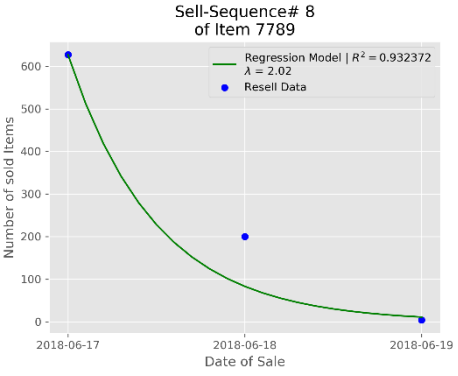


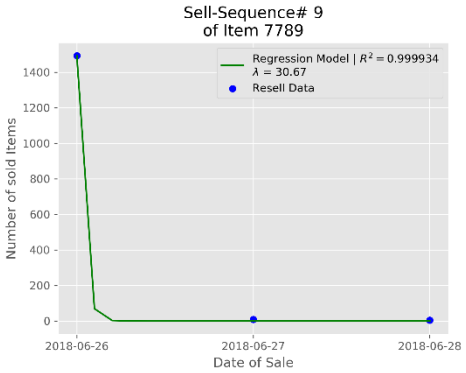




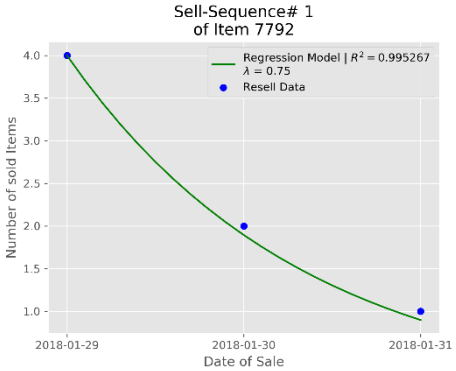


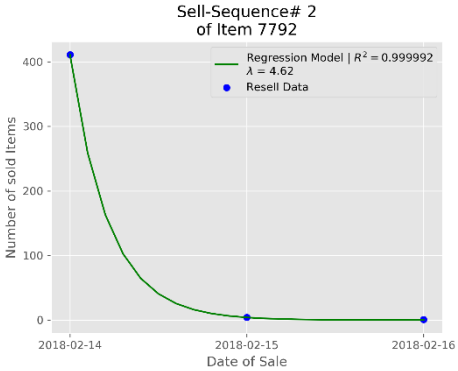


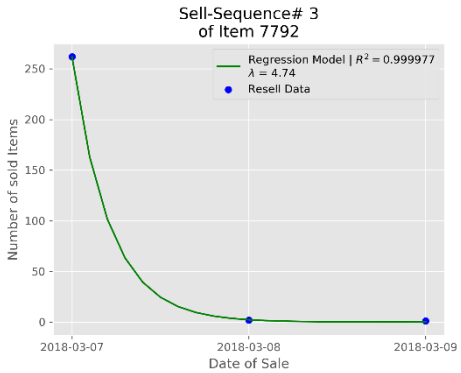


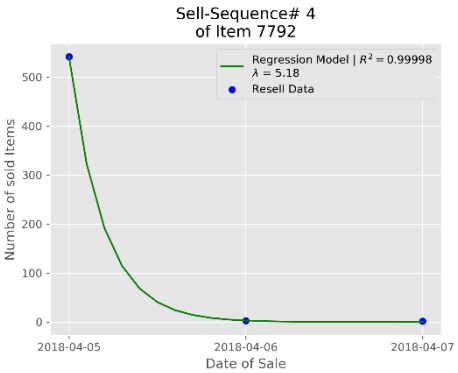


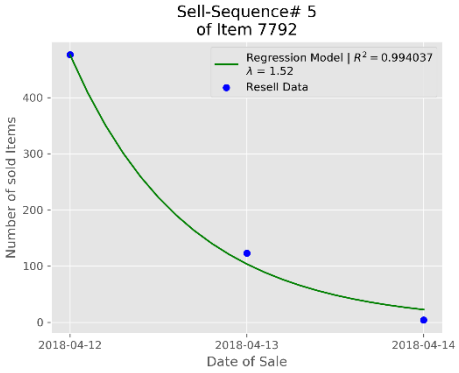
Ergebnisse der Parameterschätzung | Item_ID 7792

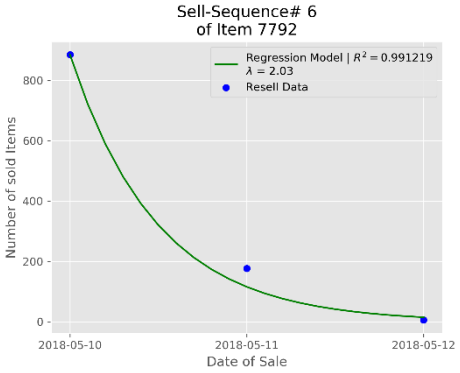


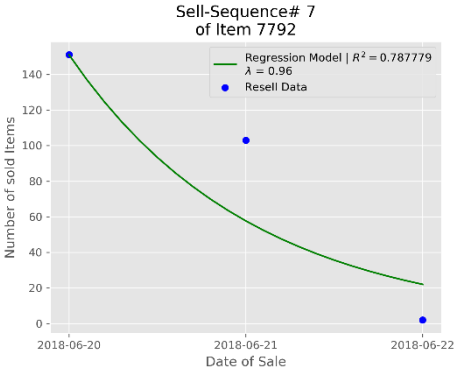


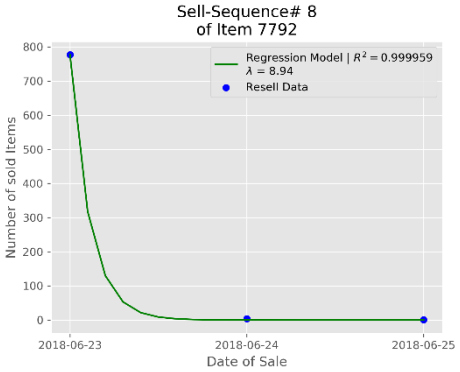




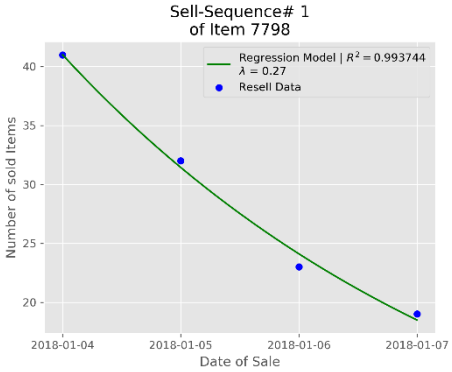


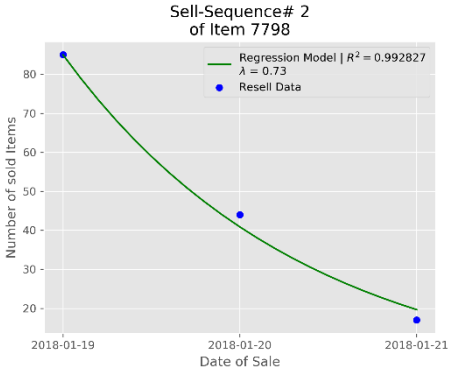


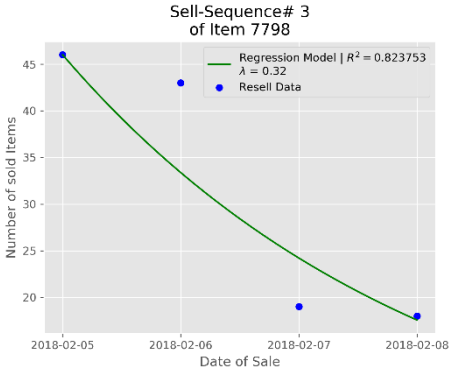


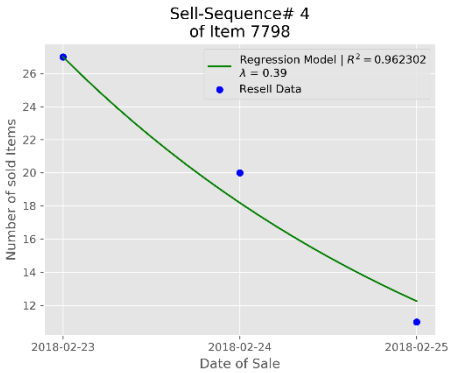


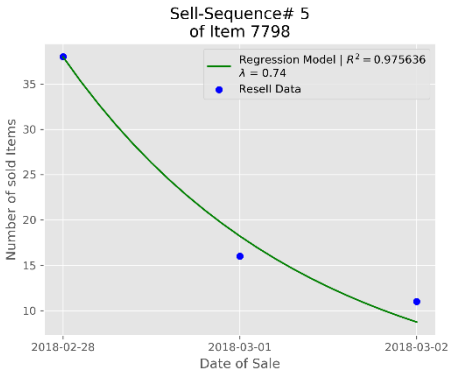
Ergebnisse der Parameterschätzung | Item_ID 7798

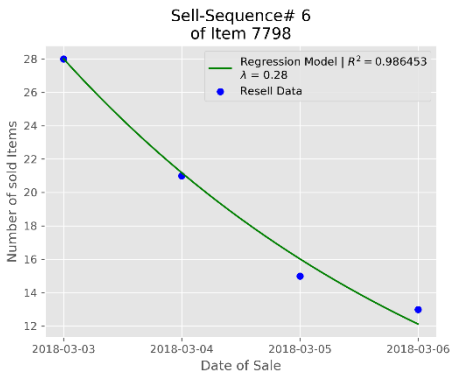


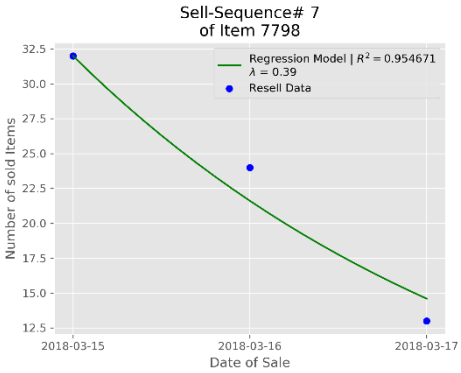


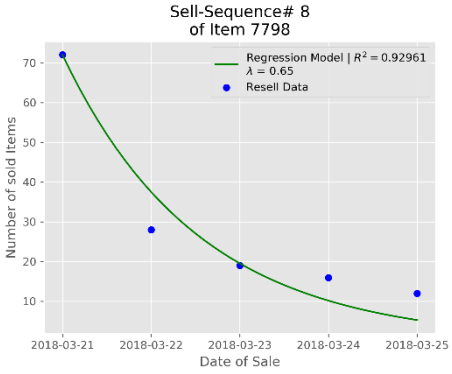

Sell-Sequence# 3
of Item 7798
Regression Model | $R^2 = 0.823753$
$\lambda = 0.32$
Resell Data
Number of sold Items
Date of Sale
2018-02-05
2018-02-06
2018-02-07
2018-02-08

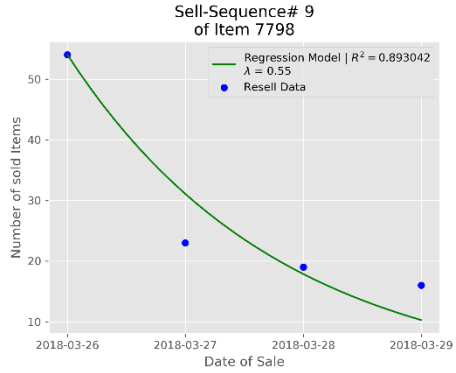

Sell-Sequence# 4
of Item 7798
Regression Model | $R^2 = 0.962302$
$\lambda = 0.39$
Resell Data
Number of sold Items
Date of Sale
2018-02-23
2018-02-24
2018-02-25

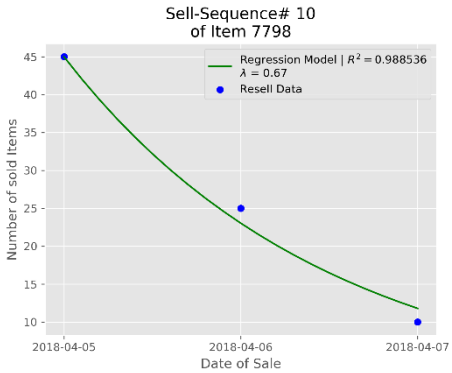

Sell-Sequence# 5
of Item 7798
Regression Model | $R^2 = 0.975636$
$\lambda = 0.74$
Resell Data
Number of sold Items
Date of Sale
2018-02-28
2018-03-01
2018-03-02

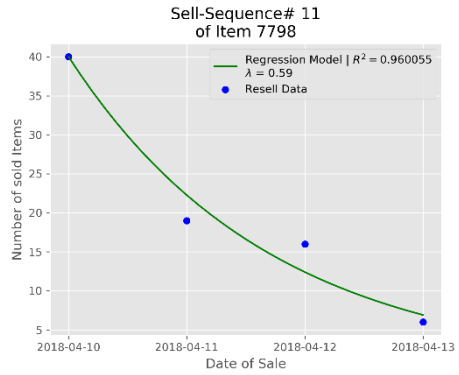

Sell-Sequence# 6
of Item 7798
Regression Model | $R^2 = 0.986453$
$\lambda = 0.28$
Resell Data
Number of sold Items
Date of Sale
2018-03-03
2018-03-04
2018-03-05
2018-03-06

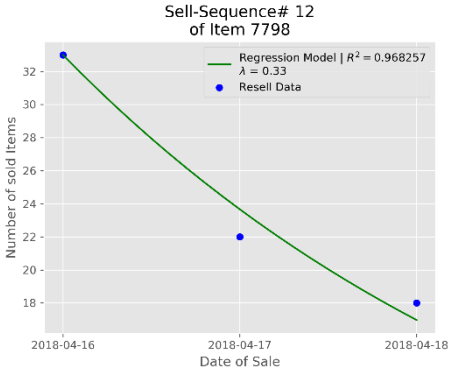

Sell-Sequence# 7
of Item 7798
Regression Model | $R^2 = 0.954671$
$\lambda = 0.39$
Resell Data
Number of sold Items
Date of Sale
2018-03-15
2018-03-16
2018-03-17

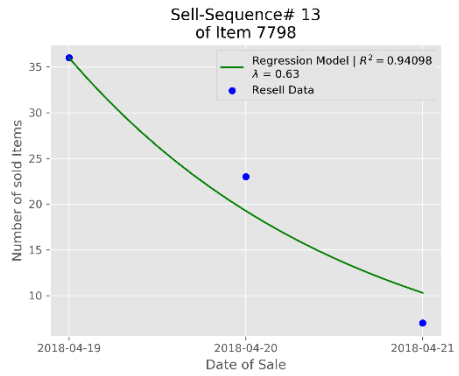

Sell-Sequence# 8
of Item 7798
Regression Model | $R^2 = 0.92961$
$\lambda = 0.65$
Resell Data
Number of sold Items
Date of Sale
2018-03-21
2018-03-22
2018-03-23
2018-03-24
2018-03-25

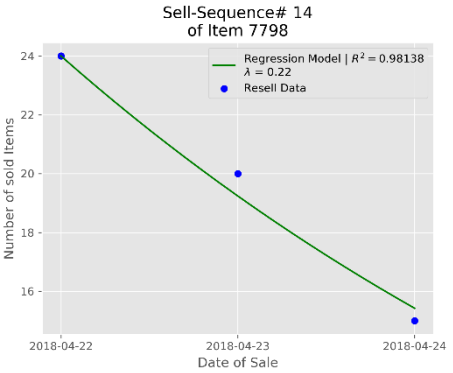

Sell-Sequence# 9
of Item 7798
Regression Model | $R^2 = 0.893042$
$\lambda = 0.55$
Resell Data
Number of sold Items
Date of Sale
2018-03-26
2018-03-27
2018-03-28
2018-03-29

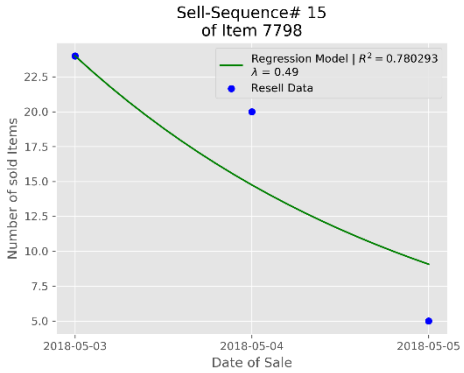

Sell-Sequence# 10
of Item 7798
Regression Model | $R^2 = 0.988536$
$\lambda = 0.67$
Resell Data
Number of sold Items
Date of Sale
2018-04-05
2018-04-06
2018-04-07

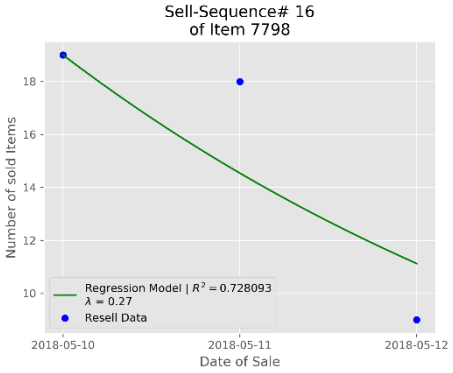

Sell-Sequence# 11
of Item 7798
Regression Model | $R^2 = 0.960055$
$\lambda = 0.59$
Resell Data
Number of sold Items
Date of Sale
2018-04-10
2018-04-11
2018-04-12
2018-04-13

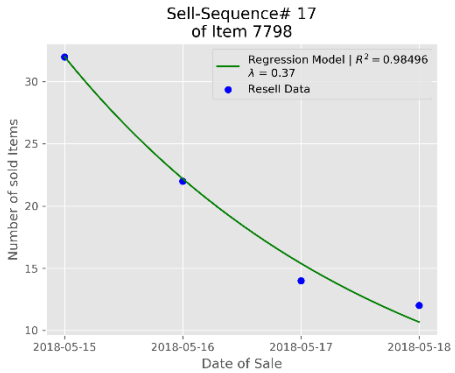

Sell-Sequence# 12
of Item 7798
Regression Model | $R^2 = 0.968257$
$\lambda = 0.33$
Resell Data
Number of sold Items
Date of Sale
2018-04-16
2018-04-17
2018-04-18

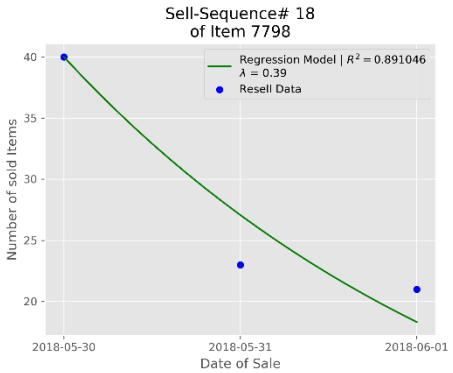

Sell-Sequence# 13
of Item 7798
Regression Model | $R^2 = 0.94098$
$\lambda = 0.63$
Resell Data
Number of sold Items
Date of Sale
2018-04-19
2018-04-20
2018-04-21

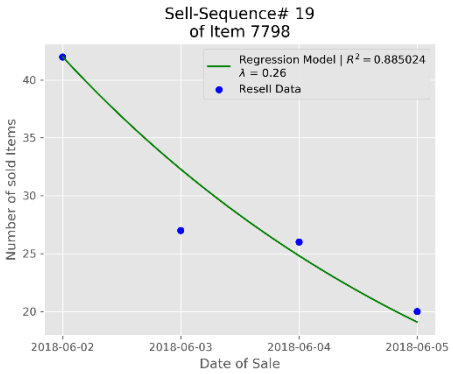

Sell-Sequence# 14
of Item 7798
Regression Model | $R^2 = 0.98138$
$\lambda = 0.22$
Resell Data
Number of sold Items
Date of Sale
2018-04-22
2018-04-23
2018-04-24

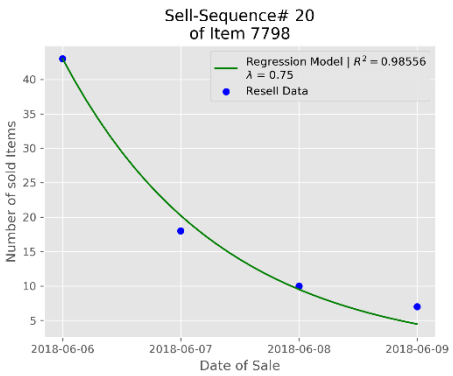

Sell-Sequence# 15
of Item 7798
Regression Model | $R^2 = 0.780293$
$\lambda = 0.49$
Resell Data
Number of sold Items
Date of Sale
2018-05-03
2018-05-04
2018-05-05

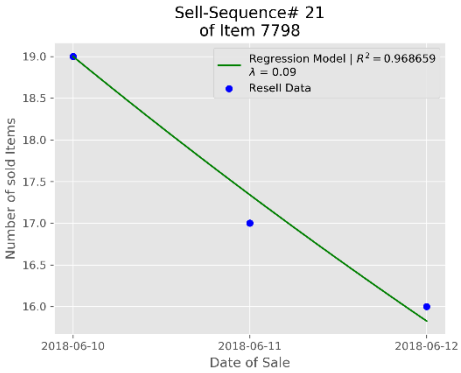

Sell-Sequence# 16
of Item 7798
Regression Model | $R^2 = 0.728093$
$\lambda = 0.27$
Resell Data
Number of sold Items
Date of Sale
2018-05-10
2018-05-11
2018-05-12

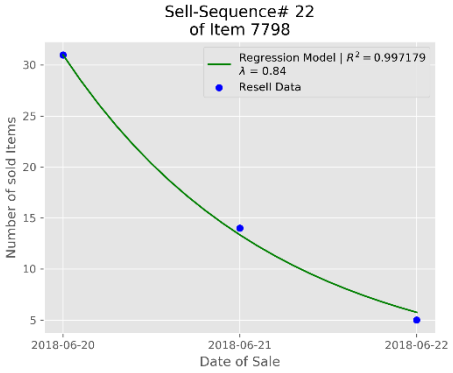

Sell-Sequence# 17
of Item 7798
Regression Model | $R^2 = 0.98496$
$\lambda = 0.37$
Resell Data
Number of sold Items
Date of Sale
2018-05-15
2018-05-16
2018-05-17
2018-05-18


Sell-Sequence# 18
of Item 7798
Regression Model | $R^2 = 0.891046$
$\lambda = 0.39$
Resell Data
Number of sold Items
Date of Sale
2018-05-30
2018-05-31
2018-06-01

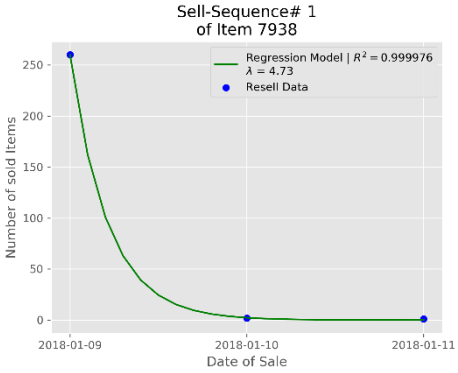


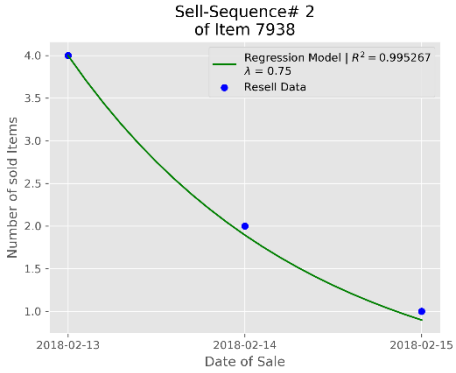


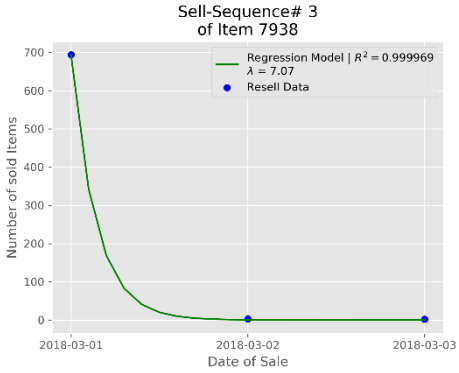


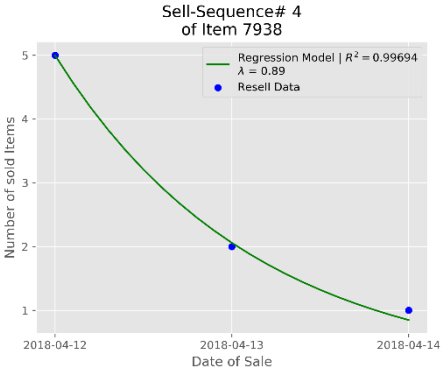


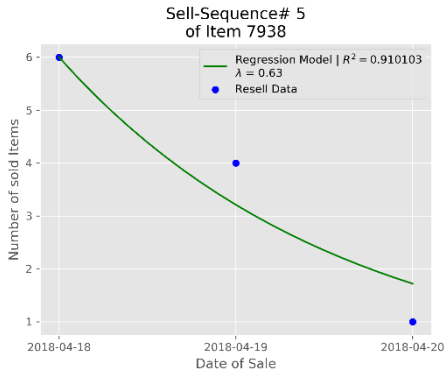
Ergebnisse der Parameterschätzung | Item_ID 7938

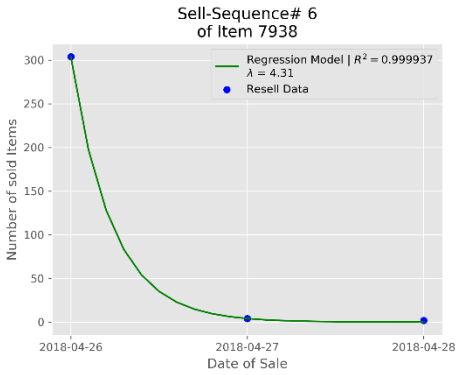


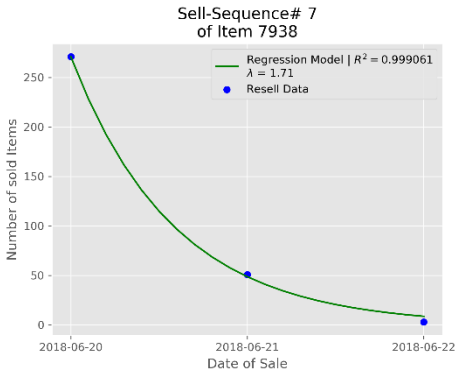


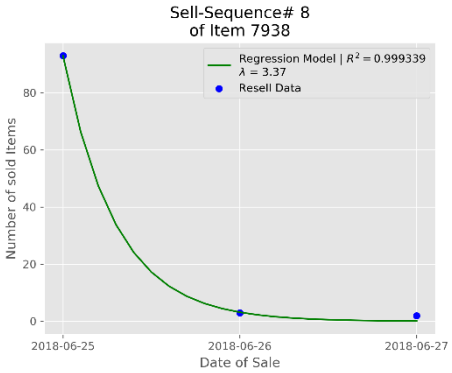




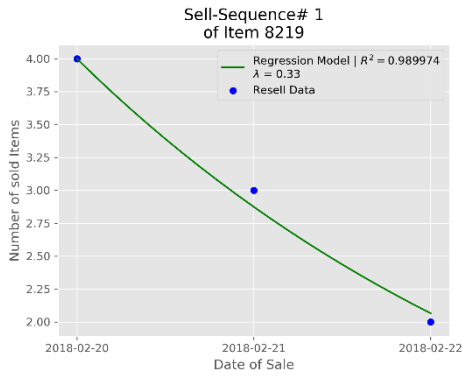


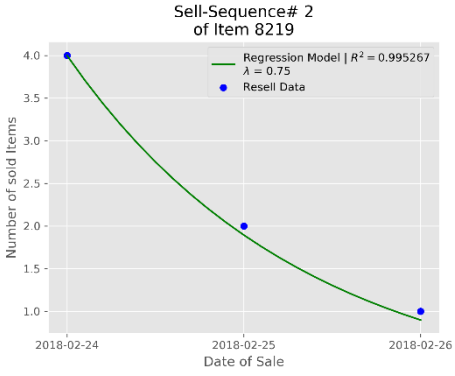


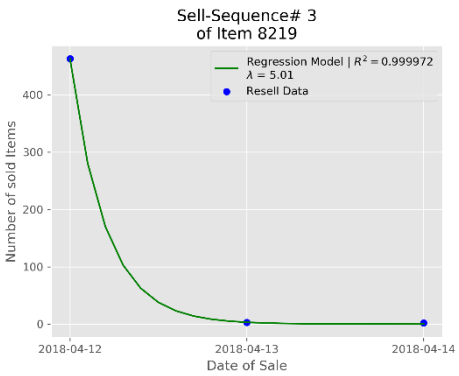


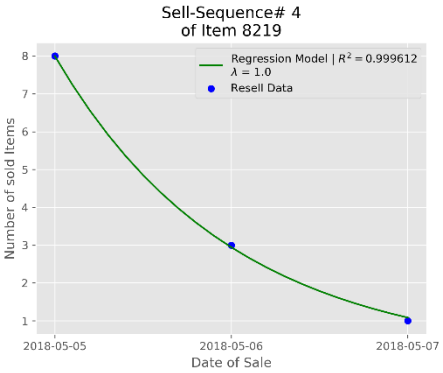


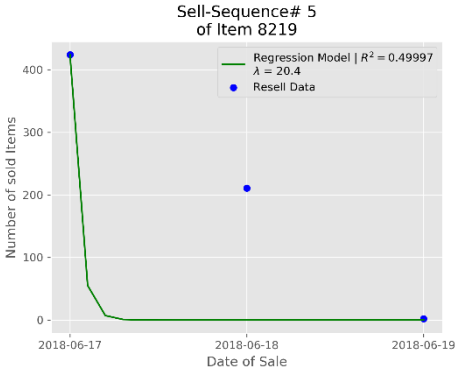
Ergebnisse der Parameterschätzung | Item_ID 7938

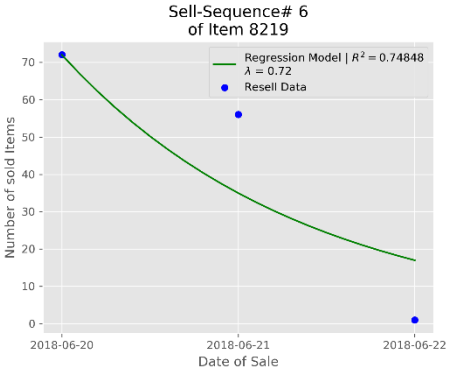


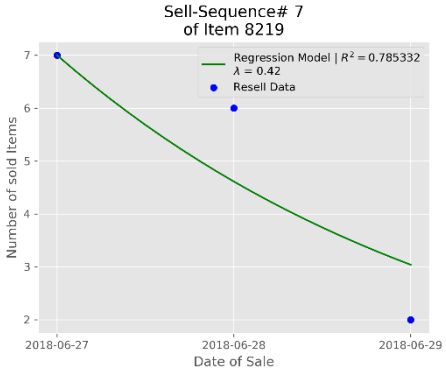


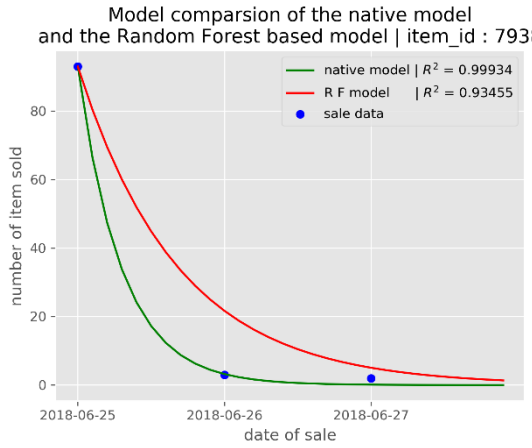

Sell-Sequence# 3
of Item 8219
Regression Model | $R^2 = 0.999972$
$\lambda = 5.01$
Resell Data
Number of sold Items
Date of Sale
2018-04-12
2018-04-13
2018-04-14

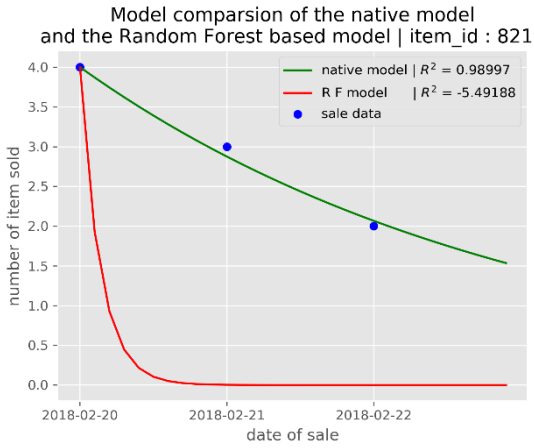

Sell-Sequence# 4
of Item 8219
Regression Model | $R^2 = 0.999612$
$\lambda = 1.0$
Resell Data
Number of sold Items
Date of Sale
2018-05-05
2018-05-06
2018-05-07

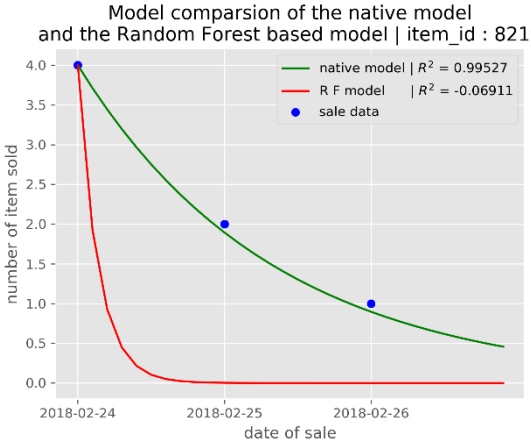

Sell-Sequence# 5
of Item 8219
Regression Model | $R^2 = 0.49997$
$\lambda = 20.4$
Resell Data
Number of sold Items
Date of Sale
2018-06-17
2018-06-18
2018-06-19

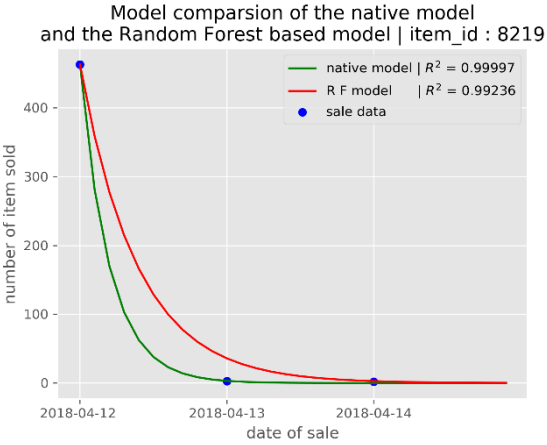

Sell-Sequence# 6
of Item 8219
Regression Model | $R^2 = 0.74848$
$\lambda = 0.72$
Resell Data
Number of sold Items
Date of Sale
2018-06-20
2018-06-21
2018-06-22

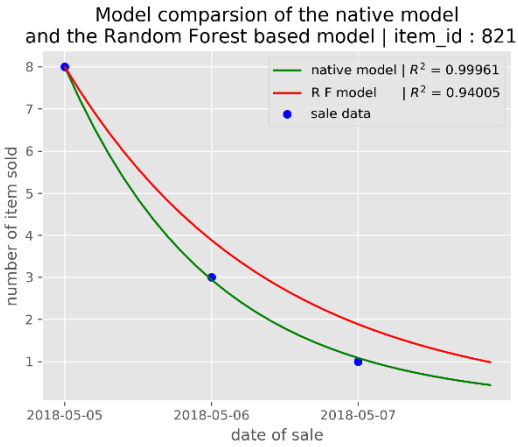

Sell-Sequence# 7
of Item 8219
Regression Model | $R^2 = 0.785332$
$\lambda = 0.42$
Resell Data
Number of sold Items
Date of Sale
2018-06-27
2018-06-28
2018-06-29

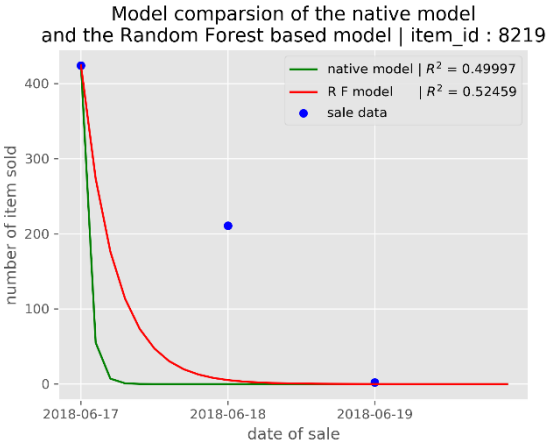

Model comparsion of the native model
and the Random Forest based model | item_id : 7938
native model | R^2 = 0.99934
R F model | R^2 = 0.93455
sale data
number of item sold
date of sale
2018-06-25
2018-06-26
2018-06-27

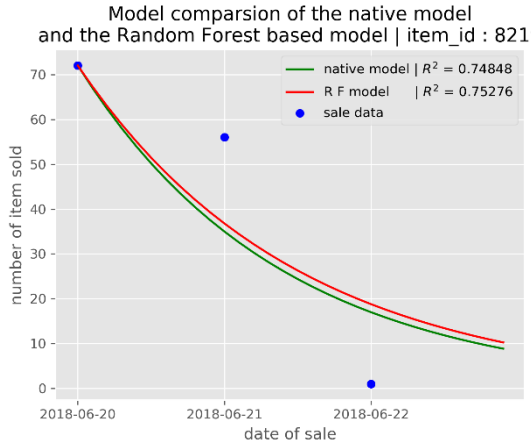

Model comparsion of the native model
and the Random Forest based model | item_id : 8219
native model | R^2 = 0.98997
R F model | R^2 = -5.49188
sale data
number of item sold
date of sale
2018-02-20
2018-02-21
2018-02-22

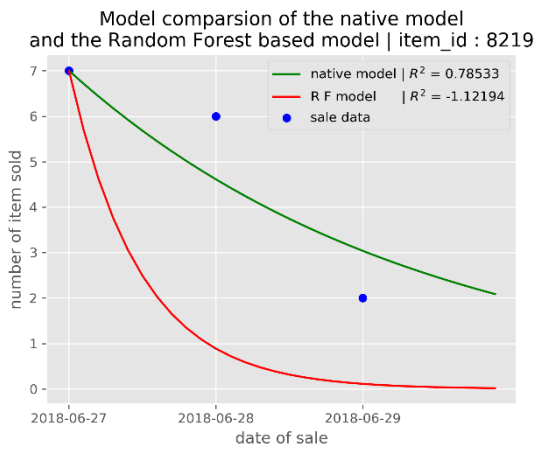

Model comparsion of the native model
and the Random Forest based model | item_id : 8219
native model | R^2 = 0.99527
R F model | R^2 = -0.06911
sale data
number of item sold
date of sale
2018-02-24
2018-02-25
2018-02-26

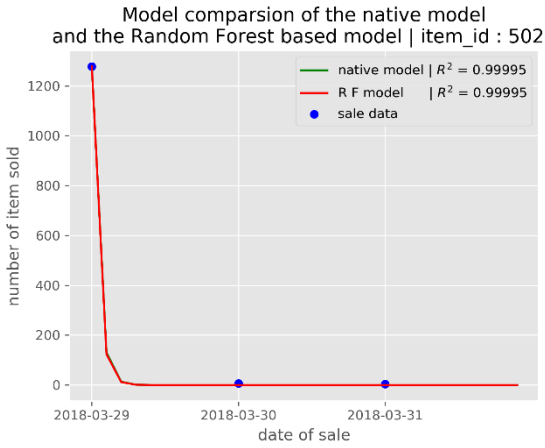

Model comparsion of the native model
and the Random Forest based model | item_id : 8219
native model | R^2 = 0.99997
R F model | R^2 = 0.99236
sale data
number of item sold
date of sale
2018-04-12
2018-04-13
2018-04-14

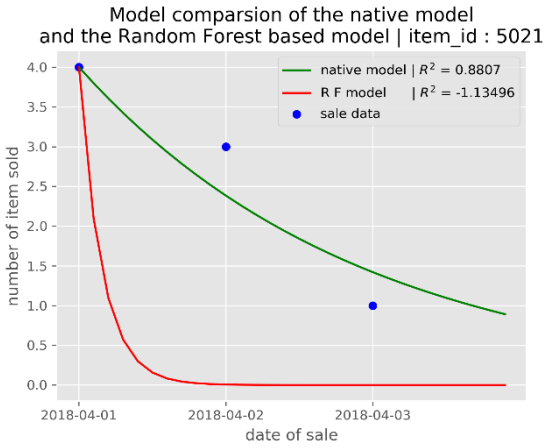

Model comparsion of the native model
and the Random Forest based model | item_id : 8219
native model | R^2 = 0.99961
R F model | R^2 = 0.94005
sale data
number of item sold
date of sale
2018-05-05
2018-05-06
2018-05-07

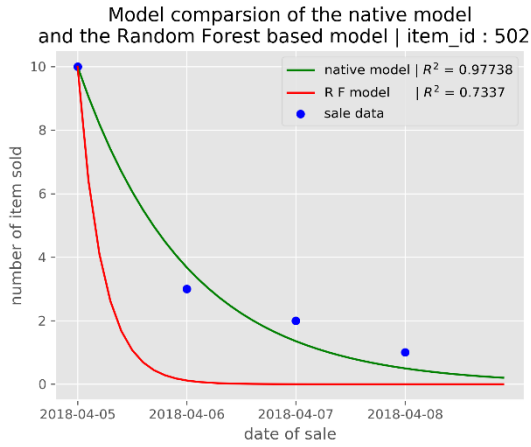

Model comparsion of the native model
and the Random Forest based model | item_id : 8219
native model | R^2 = 0.49997
R F model | R^2 = 0.52459
sale data
number of item sold
date of sale
2018-06-17
2018-06-18
2018-06-19

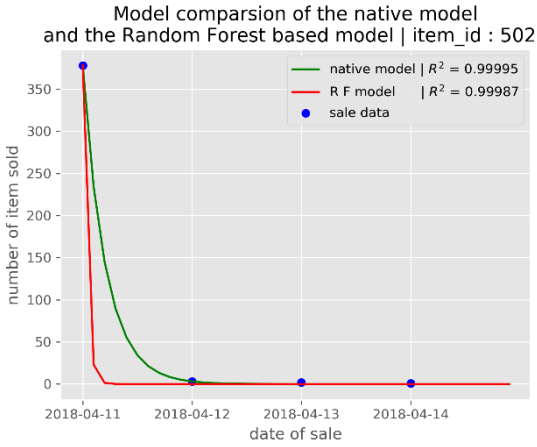

Model comparsion of the native model
and the Random Forest based model | item_id : 8219
native model | R^2 = 0.74848
R F model | R^2 = 0.75276
sale data
number of item sold
date of sale
2018-06-20
2018-06-21
2018-06-22

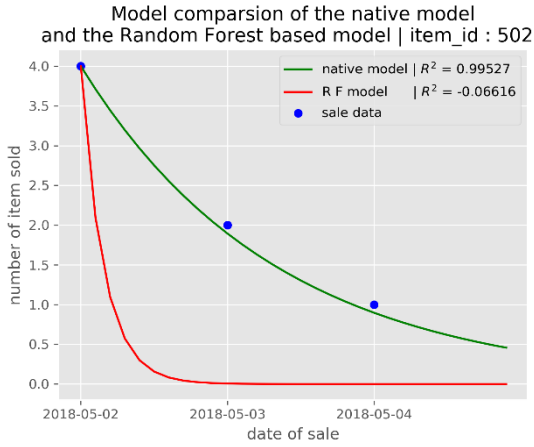

Model comparsion of the native model
and the Random Forest based model | item_id : 8219
native model | R^2 = 0.78533
R F model | R^2 = -1.12194
sale data
number of item sold
date of sale
2018-06-27
2018-06-28
2018-06-29

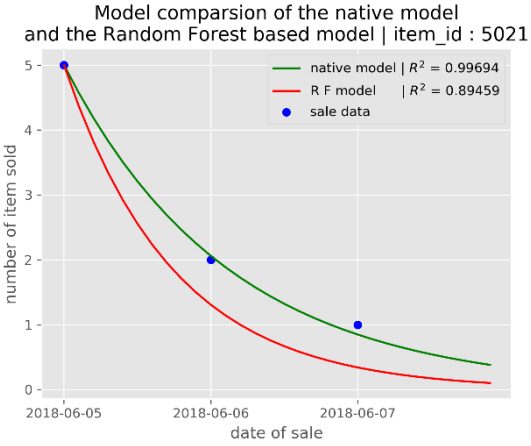

Model comparsion of the native model
and the Random Forest based model | item_id : 5021
native model | R^2 = 0.99995
R F model | R^2 = 0.99995
sale data
number of item sold
date of sale
2018-03-29
2018-03-30
2018-03-31

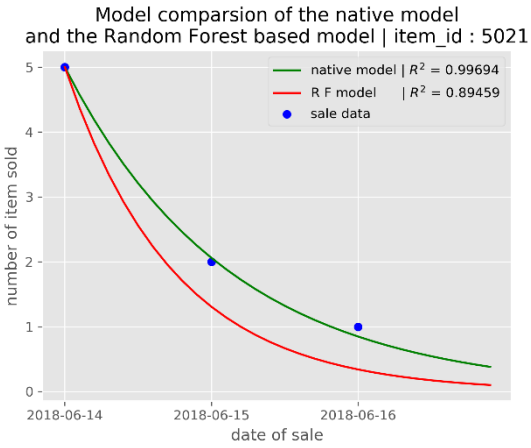

Model comparsion of the native model
and the Random Forest based model | item_id : 5021
native model | R^2 = 0.8807
R F model | R^2 = -1.13496
sale data
number of item sold
date of sale
2018-04-01
2018-04-02
2018-04-03

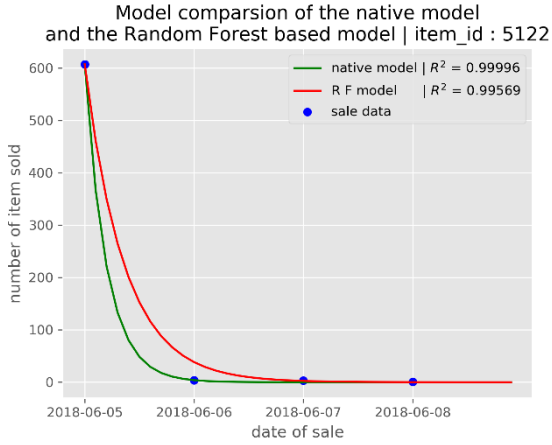

Model comparsion of the native model
and the Random Forest based model | item_id : 5021
native model | R^2 = 0.97738
R F model | R^2 = 0.7337
sale data
number of item sold
date of sale
2018-04-05
2018-04-06
2018-04-07
2018-04-08

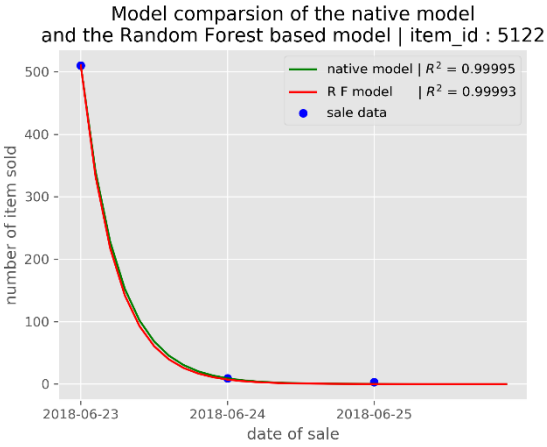

Model comparsion of the native model
and the Random Forest based model | item_id : 5021
native model | R^2 = 0.99995
R F model | R^2 = 0.99987
sale data
number of item sold
date of sale
2018-04-11
2018-04-12
2018-04-13
2018-04-14

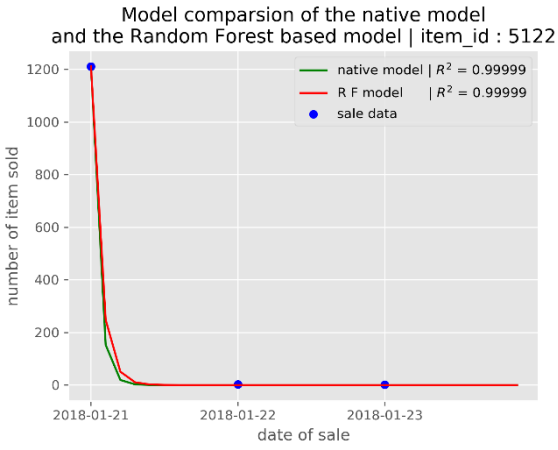

Model comparsion of the native model
and the Random Forest based model | item_id : 5021
native model | R^2 = 0.99527
R F model | R^2 = -0.06616
sale data
number of item sold
date of sale
2018-05-02
2018-05-03
2018-05-04


Model comparsion of the native model
and the Random Forest based model | item_id : 5021
native model | R^2 = 0.99694
R F model | R^2 = 0.89459
sale data
number of item sold
date of sale
2018-06-05
2018-06-06
2018-06-07


Model comparsion of the native model
and the Random Forest based model | item_id : 5021
native model | R^2 = 0.99694
R F model | R^2 = 0.89459
sale data
number of item sold
date of sale
2018-06-14
2018-06-15
2018-06-16


Model comparsion of the native model
and the Random Forest based model | item_id : 5122
native model | R^2 = 0.99996
R F model | R^2 = 0.99569
sale data
number of item sold
date of sale
2018-06-05
2018-06-06
2018-06-07
2018-06-08


Model comparsion of the native model
and the Random Forest based model | item_id : 5122
native model | R^2 = 0.99995
R F model | R^2 = 0.99993
sale data
number of item sold
date of sale
2018-06-23
2018-06-24
2018-06-25


Model comparsion of the native model
and the Random Forest based model | item_id : 5122
native model | R^2 = 0.99999
R F model | R^2 = 0.99999
sale data
number of item sold
date of sale
2018-01-21
2018-01-22
2018-01-23