Dokumentation - Modeling Toolkit

# Hintergrund und Einleitung

Das Modeling Toolkit soll als ganzheitliche Machine Learning Lösung dienen, welche sowohl visuelle Analyse- als auch Modellierungs-Funktionalitäten in einer bedienerfreundlichen App integriert. Die bereitgestellten Funktionalitäten sollen dabei eine strukturierte Vorgehensweise für die Modellierung von Sachverhalten ermöglichen. Hintergrund und Hauptanspruch des Toolkits liegt vor allem in der Demokratisierung von Machine-Learning-Methoden durch Bereitstellung eines intuitiven Benutzeroberfläche und soll nicht mit   
State-of-the-Art Verfahren im Bereich Supervised / Deep Learning konkurrieren.

# Vorabüberlegungen

Vor Verwendung der Applikation sollten zunächst Absicht und erste Hypothesen der Modellierung bedacht und formuliert werden. Folgende Fragen können hierbei als Gedankenstütze herangezogen werden:

* Was ist mein Problem bzw. Sachzusammenhang, den ich in ein Modell überführen will?
* Welche Indikatoren / Messvariablen ziehe ich generell hierfür in Erwägung? Messe ich mit den Variablen das, was meinem tatsächlichen Sachverhalt hinreichend wiederspiegelt?
* Welchen Zweck erfüllt das Modell? Dient es primär der Erklärung / statistischen Validierung des Sachzusammenhangs oder zur Vorhersage neuer Datenpunkte?
* Inwieweit sind die ausgewählten Indikatoren messbar? Sind sie hinreichend detailliert, vollständig und häufig dem Anwendungskontext gegenüber erhebbar?
* Welche Ressourcen sind mit der Modellierung bzw. dem Aufbau der notw. Dateninfrastruktur verbunden?

Mittels dieser Fragen soll ein grober Business Case für die Modellierung skizziert werden. Es sollte ein logischer Sachverhalt für eine Modellierung bestehen, der unternehmerische Mehrwerte verspricht und mit realistischen Ressourcen umsetzbar ist. Diese Vorüberlegungen sind maßgeblich, um eine praxistaugliche Modellierung und dessen Implementierung innerhalb der zugehörigen Organisation sicherzustellen.

# Nutzung zur Modellierung von zusammenhängen

Grundsätzlich kann der Modellierungsprozess in zwei Phasen aufgeteilt werden:

* Datenexplorationsphase
* Datenmodellierungsphase

## Explorative Datenanalyse

Innerhalb der Explorationsphase gilt es potenziell geeignete Erklärvariablen für die Modellierung der Zielvariable zu identifizieren und Störvariablen zu eliminieren. Desweiteren sollen Verteilungen identifiziert werden, die grundsätzliche Modellierungsannahmen verletzen und es in nachgelagerten Schritten zu transformieren gilt (z.B. Variablen mit „schiefer“ Verteilung). Je nach Modellierungstyp sollten die einzelnen Schritte unterschiedlich gewichtet werden. Auch wenn diese Funktion grundsätzlich automatisiert werden kann (z.B. via [Feature Elimination](https://medium.com/@sagar.rawale3/feature-selection-methods-in-machine-learning-eaeef12019cc)), empfiehlt es sich, „einen Überblick“ über die Daten bzw. die Verteilungen und Zusammenhänge der zugrundeliegenden Variablen zu gewinnen. Hierzu dient die *Plotting Box*, die sich im linken Bereich der Applikation befindet. Innerhalb der Box können zwei fundamentale Explorationsschritte durchgeführt werden:

* Deskriptive Häufigkeits- & Zusammenhangsanalyse
* Korrelationsanalyse

Mittels deskriptiver Häufigkeitsanalyse können eine Vielzahl von Graph-Typen (Density Plots, Bar Plots, Boxplots, Heatmaps, Scatter Plots) erstellt werden, die zur Analyse der Datenstruktur herangezogen werden können. Ähnlich eines Drag&Drop-Systems können die Variablen bzw. dessen Häufigkeiten an einer X- & Y- Variable angeordnet werden. Mittels einer dritten Grouping Variable können darüber hinaus gezielte Gruppenunterschiede innerhalb der Verteilung festgestellt werden (HINWEIS: Die Grouping Variable muss ein kategoriales Skalenniveau aufweisen). Die folgende Tabelle liefert eine Übersicht der zur Verfügung stehenden Visualisierungsoptionen:

|  |  |  |  |
| --- | --- | --- | --- |
| **Y** | **X** | **Group** | **Plot** |
| metrisch | - | Nicht vorhanden | Density Plot |
| metrisch | - | Vorhanden | Grouped Density Plot |
| metrisch | metrisch | Nicht vorhanden | Scatter Plot |
| metrisch | metrisch | Vorhanden | Grouped Scatter Plot |
| metrisch | kategorial | Nicht vorhanden | Box Plot |
| metrisch | kategorial | Vorhanden | Grouped Box Plot |
| kategorial | - | Nicht vorhanden | Bar Plot |
| kategorial | - | Vorhanden | Grouped Bar Plot |
| kategorial | kategorial | Nicht vorhanden | Heatmap |

Innerhalb des zweiten Reiters der Plotting Box sind die Zusammenhänge aller Variablen mit metrischen Skalenniveau visualisiert (kategoriale Variablen sind nicht enthalten, da hierfür keine Korrelationsanalyse möglich ist). Ziel der Korrelationsanalyse ist es Variablen mit starken (positiven als auch negativen) Zusammenhang zur Zielvariable zu identifizieren sowie multikollineare Variablenpaare (Korrelation > 0.75) für die weitere Analyse auszuschließen.

## Datenbereinigung

Zusammenfassend soll mittels der Explorationsphase ein ganzheitliches Verständnis der Daten gewonnen werden, um eine Vorab-Selektion geeigneter Modellierungsvariablen zu treffen und notwendige Pre-Processing Maßnahmen abzuschätzen. Im nächsten Schritt gilt es notwendige Transformationsschritte festzulegen, die sich aus der vorangegangenen Datenexploration ergeben – beispielsweise der Behandlung von Schiefe. Für die häufigsten Transformationsbedarfe wurden folgende Verfahren innerhalb des Preprocessing-Reiters integriert:

|  |  |
| --- | --- |
| **Standardisation** | Skaliert Mittelwert von 0 und Standardabweichung von 1 |
| **Normalisation** | Skaliert Werte zwischen 0 und 1 |
| **Log-Transformation** | Wendet den natürlichen Logarithmus auf Werte an |
| **Yeo-Johnson- // BoxCox-Transformation** | Verfahren zur Annäherung an eine Normalverteilung  und Stabilisierung der Varianz |

Da nur Variablen mit metrischem Skalenniveau zulässig für eine Modellierung sind, gilt es relevante kategoriale Variablen in ein geeignetes Format zu bringen. Mittels Encoding-Methoden werden diese hierfür in binäre Variablen (0-1) umgewandelt. Folgende Methoden wurden hierbei im Toolkit integriert:

|  |  |
| --- | --- |
| **One-Hot-Encoding** | Bildung von **n** binären Variablen bei n kategoriellen Ausprägungen |
| **Dummy-Encoding** | Bildung von **n-1** binären Variablen bei n kategoriellen Ausprägungen (letzte Ausprägung wird durch das Nichtauftreten der anderen definiert) |

Weiterhin stellt die App geeignete Funktionen zur fachgemäßen Behandlung von Outliern und fehlenden Werten bereit. Dem Anwender kann zwischen folgenden Behandlungsoptionen auswählen, die je nach Anwendungskontext durch verschiedene Vor- und Nachteile gekennzeichnet sind:

|  |  |  |  |
| --- | --- | --- | --- |
|  | *Skalenniveau* | *Funktion* | *Beschreibung* |
| **Fehlende Werte:** | Metrisch | ***Delete*** | Beobachtung mit fehlenden Werten löschen |
|  | Metrisch | ***Median*** | Ersetzen fehlender Werte mit Median der Variable |
|  | Metrisch | ***Mean*** | Ersetzen fehlender Werte mit Mittelwert der Variable |
|  | Diskret | ***Keep*** | Beobachtung mit fehlenden Werten löschen |
|  | Diskret | ***Mode*** | Ersetzen fehlender Werte mit Modus der Variable |
|  | Diskret | ***Random*** | Ersetzen fehlender Werte mit zufälliger Ausprägung |
|  |  |  |  |
| **Outlier:** | Metrisch | ***Keep*** | Keine Behandlung von Outliern |
|  | Metrisch | ***Median*** | Ersetzen von Outliern durch Median |
|  | Metrisch | ***Mean*** | Ersetzen von Outliern durch Mittelwert |

Nicht zuletzt bietet die Applikation diverse Filter-Funktionen, um Variablen mit Eigenschaften zu entfernen,   
die eine Beeinträchtigung - wenn nicht sogar Verhinderung – des Trainingsprozesses zur Folge haben können:

|  |  |
| --- | --- |
| **Multikollinearität** | Starke Korrelation zwischen Prädiktoren |
| **Linearkombination** | Prädiktoren stellt Linearkombination eines anderen Prädiktors dar |
| **Varianzhomogenität** | Variable weist eine Varianz von (nahezu) 0 auf |
|  |  |

## Modellierung

Nachdem der Modellierung vorangehende Transformationsmaßnahmen definiert wurden, gilt es den zu verwendenden Modellierungsalgorithmus (bzw. -algorithmen) auszuwählen.Grundsätzlich basiert die Entscheidung, wie viele Modelltypen gleichzeitig trainiert werden sollen, auf einem Trade-Off zwischen Performance und Rechenleistung. Dabei ist meist die Datengröße das ausschlaggebende Kriterium für die Abwägung dieses Trade-Offs. Mit zunehmender Datenmenge steigt zwar meist die Genauigkeit bzw. die Generalisierbarkeit des Modells, allerdings steigt hiermit auch die Zeit- & Rechenaufwand pro Trainingsperiode. Als Faustregel empfiehlt es sich auf Basis der unten dargestellten Tabelle eine Auswahl an 2-3 Algorithmen zu treffen, die für die spezifische Problemstellung potenziell erfolgreich sein könnten. Bei extrem großen Datenmengen (Datensätzen > 1 TB) empfiehlt es sich, lediglich ein Algorithmus pro Trainingsperiode einzubeziehen. Folgende Algorithmen stehen zum aktuellen Stand zur Verfügung:

|  |  |  |
| --- | --- | --- |
| **lm** | Linear Regression Algorithmus | Regression |
| **lda** | Linear Discriminant Analysis | Classification |
| **ada** | Ada Boosting Algorithmus | Regression, Classification |
| **rf** | Random Forest Algorithmus | Regression, Classification |

Eine ausführlichere Dokumentation der Algorithmen inklusive ihrer Vor-& Nachteile der verschiedenen Modelle ist unter [folgendem Link](https://topepo.github.io/caret/available-models.html) zu finden. Als „Allrounder“ haben sich hierbei sog. Ensemble-Algorithmen (wie rf, ada) bewährt, welche durch eine extrem hohe Flexibilität im Anwendungsbereich bei einer gleichzeitig hohen Genauigkeit gekennzeichnet sind. Nach abschließender Auswahl der Modellierungsalgorithmen gilt es die Vorgehensweise innerhalb des Trainingsprozesses genauer zu spezifizieren. Ein entscheidender Schritt hierbei ist die Auswahl des Indikators, nach dem das Modell optimiert werden soll. Dabei stehen dem Anwender unterschiedliche Indikatoren pro Problemstellung zur Verfügung:

|  |  |  |
| --- | --- | --- |
|  | *Problem-Typ* | *Erläuterung* |
| [**Accuracy**](https://towardsdatascience.com/20-popular-machine-learning-metrics-part-1-classification-regression-evaluation-metrics-1ca3e282a2ce) | Klassifikation | # korrekte Vorhersagen / # Vorhersagen |
| [**Cohen’s (unweighted) Kappa**](https://thedatascientist.com/performance-measures-cohens-kappa-statistic/) | Klassifikation | Statistik zur Überprüfung d. Interrate-Reliability |
| **[R²](https://towardsdatascience.com/statistics-for-machine-learning-r-squared-explained-425ddfebf667)** | Regression | Erklärte Varianz / Gesamt-Varianz |
| [**RMSE (Root mean squared error)**](https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234) | Regression | Standardabweichung der Residuen |

Ein weiterer, relevanter Aspekt ist das „Tuning“ von Hyperparametern – sprich von jenen Parametern, die der Algorithmus nicht selbst optimieren kann und daher vom Menschen ausgewählt werden müssen. In der Praxis hat es sich bewährt, diesen Auswahlprozess einem klassischen Trial-&-Error Ansatzes folgen zu lassen, welcher innerhalb des Trainingsprozesses automatisiert wird. Dabei wurden gängige Verfahren in der App intergiert:

|  |  |
| --- | --- |
| **Random Hyperparameter Search** | Zufällige Auswahl von Parameterwerten aus einem abgesteckten Wertebereich |
| **Grid Hyperparameter Search** | Festlegung von spezifischen Parameterwerten |

Sollte sich für die Random-Ansatz des Hyperparameter Tunings ausgewählt werden, erscheinen innerhalb der Toolkit-Sidebar entsprechende Slider zur Auswahl der einzubeziehenden Parameterbereiche. Aufgrund der Komplexität dieser Parameter sollte diese nur in Sonderfällen in Erwägung gezogen werden. Die dritte Entscheidungskomponente innerhalb der Trainings-Spezifizierung ist die Definition der Train & Test-Set Größe. Dieses Verfahren beruht auf dem gängigen Prinzip innerhalb der Data Science Praxis, den vorliegenden Datensatz in zwei distinktive Komponenten zu unterteilen. Das sogenannte Train-Set dient der Optimierung des Modells. Das Test-Set wiederum ist eine Art Versicherung für die Praxistauglichkeit für das „trainierte Modell“. Es wurde vom Algorithmus bislang nicht „gesehen“ und spiegelt daher eine gute Vergleichbarkeit zu neuen Daten wider, die später auch in der Praxis zur Klassifikation herangezogen werden sollen. Das übliche Verhältnis von „80% Training - 20% Testing“ ist standardgemäß innerhalb der Applikation voreingestellt und kann nach Belieben adaptiert werden. Eine weitere, naheliegende Determinante liegt in der Auswahl des Validierungsprozesses, der im Rahmen des Hyperparameter-Tunings angewendet wird. Zum aktuellen Stand kann aus folgenden Validierungsverfahren ausgewählt werden:

|  |  |
| --- | --- |
| **boot** | Bootstrapping |
| **cv** | K-Fold Cross Validation |
| **repeatedcv** | Repeated K-Fold Cross Validation |
| **LOOCV** | Leave-One-Out Cross Validation |
| **LGOCV** | Leave-Group-Out Cross Validation |

Bei allen genannten Verfahren bedarf es einer Einstellung der Resampling-Iterationen – sprich der Anzahl, wie häufig das Ergebnis validiert werden soll. Ähnlich zur Anzahl der einbezogenen Modellalgorithmen vermehrt sich die Kalkulationsdauer mit jeder Validierungsschleife. Die Auswahl sollte daher wohlüberlegt und verhältnismäßig zu der Datengröße bzw. den Genauigkeitsansprüchen des Kunden abgewogen werden. Einzelne Validierungsverfahren bieten darüber hinaus noch granulare Einstellungsmöglichkeiten. Eine ausführliche Erläuterung der Verfahren und ihrer Parameter kann unter [folgendem Link](https://towardsdatascience.com/validating-your-machine-learning-model-25b4c8643fb7) gefunden werden.

Nach Bereinigung des Datensatzes sowie der Auswahl der Trainingsparameter und einzubindenden Algorithmen kann die Modellierung gestartet werden. Da bereits kleine Fehler die Trainingsdauer ungeplant verlängern können, empfiehlt es sich, vorab einen letzten Sanity Check zur Überprüfung der Richtigkeit der Parameter durchzuführen. Während des Trainingsprozesses erscheint unten rechts ein Status-Balken, der eine grobe Indikation bezüglich der Fortschreitung des Trainingsprozesses dient. Da leider keine konkretere Zeitangabe darstellbar ist, empfiehlt es sich, iterative Modellierungsperioden mit steigender Komplexität (etwa durch mehr Validierungsschritte oder Einbezug mehrerer Algorithmen) durchzuführen, um ein grobes Gefühl für den „Zeitaufwand“ zu bekommen. Somit kann sichergestellt werden, dass nicht mehr Zeit und Rechenleistung investiert wird als zwingend notwendig. Sollte ein Trainingsprozess unerwartet lange dauern, lässt sich dieser in der Regel durch Schließung der App abbrechen. In seltenen Fällen kann es dazu kommen, dass die Applikation dadurch nicht geschlossen wird und im Hintergrund weiter aktiv ist. Daher ist es ratsam, die tatsächliche Schließung des Programmes im Task Manager zu überprüfen.

## Modell-Bewertung

Nachdem das Training der ausgewählten Modelloptionen abgeschlossen ist, kann zur Performance-Analyse fortgeschritten werden. Hierbei ist die „Performance Box“ im rechten Teil des Dashboards entscheidend.   
Der erste Reiter dient als modellübergreifende Übersicht, in dem alle für die Problemstellung relevanten Indikatoren je Algorithmus dargestellt sind. Im Reiter „Model Metrics“ sind wiederum genauere Performance-Informationen zu den einzelnen Modellen aufgelistet, welche im Folgenden genauer erläutert werden:

|  |  |  |
| --- | --- | --- |
|  | *Problem-Typ* | *Erläuterung* |
| [**Confusion Matrix**](https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234) | Classification | Siehe Link |
| [**Group-specific Metric Plot**](https://towardsdatascience.com/20-popular-machine-learning-metrics-part-1-classification-regression-evaluation-metrics-1ca3e282a2ce) | Classification | Siehe Link |
| **Density Comparison Plot** | Regression | Prognostizierte VS Tatsächliche Verteilung |
| [**Residual Plot**](https://towardsdatascience.com/how-do-you-check-the-quality-of-your-regression-model-in-python-fa61759ff685) | Regression | Siehe Link |

Die beiden übrigen Reiter enthalten Indikatoren bezüglich der Modell-Interpretierbarkeit. Eine genauere Erläuterung [globaler](https://towardsdatascience.com/hands-on-global-model-interpretation-3bb4264732b5) und [lokaler](https://gilberttanner.com/blog/local-model-interpretation-an-introduction) Interpretationsmethoden von Modellen findet sich unter den angehängten Links. Ein wesentlicher Unterschied zwischen beiden Methoden besteht darin, dass globale Methoden den eingeflossenen Variablen einen allgemeine Bedeutungswert für die Entscheidungsfindung zuschreiben, wohingegen lokale Methoden sich auf die Entscheidungsmechanismen und die Variableneinfluss bei einzelnen Ausprägungen fokussieren. Innerhalb des Reiters „Variable Importance“ ist der allgemeine Erklärungsbeitrag der Variable visualisiert, was als globaler Interpretationsindikator in Erwägung gezogen werden kann. Innerhalb des Reiters „SHAPLEY Values“ können die Entscheidungsmechanismen des Modells für die ersten 100 Fälle des Test-Sets visualisiert werden. Diese lassen sich aus den dargestellten SHAPLEY-Werten des zu begutachtenden Falles ableiten und dienen als lokaler Interpretationsindikator. Innerhalb von Regressions-Modellen lassen sich diese als (positiver/negativer) Einfluss auf den Wert der Zielvariable interpretieren. Für Klassifikationsmodelle, insbesondere jene mit multinomialer Natur (mehr als 2 zu prognostizierende Kategorien), gilt eine abweichende Interpretation der SHAPLEY-Werte. Hierbei lassen sich diese näherungsweise als (positiver /negativer) Einfluss auf die Wahrscheinlichkeit der jeweiligen Ausprägung interpretieren und sind daher für jede Klasse einer kategorialen Zielvariable zu begutachten.

Mit Ausnahme des Übersichts-Reiters lassen sich die Graphiken für alle einbezogenen Algorithmen begutachten. Die Auswahl des zugehörigen Modells erfolgt hierbei durch das entsprechende Auswahlfeld im Header der Applikation. Neben dem Reiter werden die Performance-Indikatoren des ausgewählten Modells durch die vier oberen KPI-Boxen zusammengefasst.

## Modell-Auswahl und -Export

Die visualisierten Graphiken, insbesondere innerhalb der „Performance“-Box sollen einen ganzheitlichen Überblick über die Qualität der einzelnen Modell-Optionen vermitteln. Dabei liefern die ersten beiden Reiter sowie die KPI Boxen einen detaillierten Einblick in die Prognosegenauigkeit der Optionen, wohingegen die letzteren Reiter einen tieferen Einblick in die Prognosemechanismen des Modells geben. Sie zusammen sollen das Informationsfundament bilden, um Modell-Interpretierbarkeit und -genauigkeit sorgfältig im Rahmen des jeweiligen Projektkontextes abwägen und ein präferierten Modells auswählen zu können.

Nach erfolgreicher Entscheidungsfindung bietet das Modeling Toolkit verschiedenste Möglichkeiten zur praxisbezogenen Weiterverwertung des Modells. So bietet es zum einem die Möglichkeit, das Modell sowie vorangegangene Pre-Processing Schritte durch Klicken des „Download Model“-Buttons in einer .zip-Datei zu exportieren. Dabei werden die Daten in das „.rds“ Format transformiert, welches von allen R-Programmen gelesen werden kann. Mittels des zugehörigen Skriptes „XXX.R“ können neue Werte problemslos klassifiziert werden. Hierzu gilt es zuvor den Dateipfad zum Modell und Pre-Processing Schritten sowie den neuen Datensatz innerhalb des Skriptes anzupassen. Darüber hinaus besteht die Möglichkeit, das exportierte Modell in ein interaktives „Prediction Tool“ einzubetten, welches ähnlich zum Modeling Toolkit selbst eine intuitive Benutzeroberfläche bietet. Das Prediction Tool ermöglicht das Öffnen eines Datensatzes (via Datenbank oder Datei-Upload) und klassifiziert die Daten automatisch mit Hilfe des eingebetteten Modelles. Eine Analyse der Varriablenverteilungen sowie der Prognosemechanismen (via SHAPLEY-Werten) kann unter den entsprechenden Reitern vorgenommen werden. Der Export des prognostizierten Datensatzes innerhalb des Prediction Tools erfolgt via CSV-Export.

Weiterhin kann durch Klicken des „Download Report“ Buttons ein automatisierter Modell-Report mit interaktiven Graphiken heruntergeladen werden. Hierbei muss der Anwender unter dem Pop-Up Menü lediglich den Projektnamen, -hintergrund sowie die Begründung für die Modellentscheidung spezifizieren.

# Überführung des Modells in das Prediction-Toolkit

Die Überführung in das Prediction Toolkits erfolgt nach einem standardisierten Prozess, der im Folgenden genauer aufgelistet ist:

1. Kopiere den Template-Ordner und benenne ihn entsprechend des Projektkontextes um
2. Ändere den konkreten Namen des Toolkits sowie die zugehörige Farbpalette (bestehend aus Hauptfarbe und Sekundärfarben, im HEX-Format) innerhalb der „ui.R“-Datei in den zugehörigen Variablen ab. Da ein Großteil des User Interfaces in der Hauptfarbe gestaltet ist, sollte hierfür eine möglichst assoziationsstarke Farbe – wie etwa die Corporate Color – verwendet werden. Die Sekundärfarben werden innerhalb der Graphen – speziell in der Visualisierung von Gruppen-unterschieden – genutzt und sollten damit hinreichend unterscheidbar sein. Die Anzahl der notwendigen Sekundärfarben richtet sich meist an der kategoriellen Variable mit den meisten Ausprägungen innerhalb des Datensatzes, der zur Modellierung herangezogen wurde. Als grobe Faustregel sollten mindestens 4 unterscheidbare Sekundärfarbe eingesetzt werden.
3. Sollte ein Logo-Branding gewünscht sein, füge eine entsprechende „logos.png“ und „logos.ico“ Datei  
   in den Subordner „WWW“ des Template-Ordners. Das PNG wird oben rechts innerhalb des Toolkits und als ICO als generelles Logo der Applikation verwendet. Auf der [folgenden Website](https://www.zamzar.com/convert/png-to-ico/) können PNGs einfach in das entsprechende Icon Format konvertiert werden.
4. Füge die zugehörige ZIP-Datei des exportierten Modells Toolkit beruht auf den Modellen in den Template Ordner ein.
5. Nun kann das Datei in eine Windows-Installationsdatei überführt werden. Hierzu muss die R.Datei „prediction\_tool\_deployment.R“ geöffnet werden. Zunächst gilt es die Zeilen 1-3 zu markieren und auszuführen (STRG + ENTER). Hierbei werden notwendige Packages installiert und aktiviert. In den Zeilen X-X sind wichtige Parameter der Installationsdatei, die je nach Projektkontext angepasst werden können (z.B. Umbenennen der Applikation oder Änderung der Administrationsbefugnisse):

|  |  |  |
| --- | --- | --- |
| **app\_name** | STRING | Name of the Application |
| **app\_dir** | STRING | Directory of the App Folder (Update Accordingly) |
| **include\_R** | BOOLEAN | Indicate if Installer should contain a version of R (Required for all PCs without R installed) |
| **privilege** | STRING | Setup Installation Requirements - “high”: Administration Status required - “low”: Everyone can install Application |
| **default\_dir** | STRING | Default Installation Directory for Application (see XXX for further details) |
| **app\_icon, setup\_icon** | STRING | Name of Application / Setup Icon  (suggested to keep naming and overwrite “app\_icon.ico” with a custom icon the main folder) |
| **pkgs** | VECTOR | List of required packages to launch the App (If no major changes to the app are applied, keep the list) |
| **Electron Dings** |  | VS FIREXFOX ODER CHROME |

1. Nachdem die Parameter für die Applikation angepasst wurden und das Programm „INNO“ installiert wurde, kann man durch den Befehl „runApp(…)“ sicherstellen, dass die Applikation erwartungsgemäß funktioniert und keine Fehler gemacht wurden. Nach erfolgreicher Prüfung der Applikation, kann schließlich der Befehl „compile\_iss()“ ausgeführt werden. Hierdurch wird im ausgewählten Verzeichnis „app\_dir“ die Dateien bereitgestellt, die zur Installation der Programmes notwendig sind. Dieser Prozess kann einige Minuten dauern.
2. Nach Abschluss der Prozesses kann die App auf allen aktuellen Windows PCs installiert werden. Die zugehörige Installations-Datei befindet sich im Sub-Ordner „RInno\_installer“. Wie vorab definiert, sollten die Lizenztexte sowie der Standard-Installationspfad erwartungsgemäß angezeigt werden.
3. Nach erfolgreicher Installation kann die App geöffnet werden. Bei erster Ausführung der Applikation kann sich die Ladezeit aufgrund dem einmaligen Laden von Paketen etwas verzögern. In der Folgenutzung sollte dies aber nicht mehr der Fall sein.

# Troubleshooting

Generell kann der Auftritt von Fehlern innerhalb des Deployment-Prozesses auch bei anleitungsgemäßer Erstellung nicht gänzlich ausgeschlossen werden. Je nach Fehlerherkunft ist dessen Behebung trivial oder erfordert spezielles IT-Vorwissen. Zur präventiven Vermeidung solcher Fehler empfiehlt es sich daher, sich an der obenstehenden Anleitung bestmöglich zu orientieren.

Zur Eingrenzung der Fehler sollte zunächst die generelle Funktionsfähigkeit der Applikation mittels des „runApp“- Befehls des Deployment-Skriptes überprüft werden. Sollten hier bereits Fehlermeldung auftreten, scheint bei der Abänderung der App-Parameter ein Fehler passiert zu sein. Typische Fehler in dieser Phase sind:

* Ein Sonderzeichen zu viel / wenig (z.B. . , / ( ) \* # )
* Falsche Schreibweise der Farben (siehe folgenden Link für Hexagon Schreibweise)
* Falsche Benennung der .ico und .png Dateien
* Updates innerhalb der geladenen Packages, die zu einer veränderten Nutzungsweise einzelner Funktionen führen
* Sonstige Fehler, die durch aktive Abänderung & Erweiterung des Codes eingingen

Beim Auftreten eines solchen Fehlers sollte zunächst die zugehörige, rot markierte Fehlermeldung innerhalb von R begutachtet werden. Wird daraus die Fehlerquelle nicht unmittelbar erschließbar, empfiehlt es sich, diese im Ganzen oder in Teilen in einer Suchmaschine einzugeben und in Foren nach einer Erläuterung zu suchen. Hierbei haben sich vor allem die Plattformen „Stack Overflow“ sowie „Github“ bewiesen. Sollte die Applikation normal angezeigt werden, kann die Roh-Applikation bzw. dessen zugrundeliegender Code mit großer Wahrscheinlichkeit als fehlerfrei eingestuft werden.

Zur weiteren Fehlerdiagnose sollte die Funktionalität des Deployment-Packages „RInno“ überprüft werden. Es dient dazu, die Roh-Applikation (bestehend aus den zwei .R-Dateien „ui.R“ und „server.R“) in einer klassische Installationsdatei zu überführen. Nachstehend sind mögliche Fehlerquellen und Behebungsansätze innerhalb des Deployment-Prozesses aufgelistet:

* Inno was not installed (properly) 🡪 Please deinstall and reinstall the latest Inno Version
* Falsche Verzeichnis- und / oder Ordnerangabe
* „server.R“ oder „ui.R“ sind während des Deployment-Prozesses noch geöffnet  
  (oder noch im Zwischenspeicher einer RStudio Session gespeichert)

Die Liste erhebt keinen Anspruch auf Vollständigkeit, was bedeutet, dass in Zukunft weitere, unbekannte Fehler auftreten können. Es empfiehlt sich daher bei Auftreten eines unbekannten Fehlers die [Troubleshooting Seite](https://github.com/ficonsulting/RInno/issues) von „RInno“ bzw. der anderen „Package“-Urheber zu sichten und nach passenden Lösungsoptionen zu überprüfen. Sollte man hier keine zufriedenstellende Antwort finden, sollte das Melden des Fehlers in Erwägung gezogen werden. Dies erfordert lediglich eine kostenfreie Anmeldung unter Github und wenigen Klicks unter dem Reiter „File an Issue“. Urheber selbst haben meist ein großes Interesse an der Meldung von Fehlern und reagieren auf diese in der Regel sehr schnell und zuvorkommend.

# Limitationen

Aufgrund der bedingten Ressourcen bietet dem Tool zusätzliche Feature-Ideen, die bislang noch nicht integriert werden konnten. Im Folgenden sollen diese aufgelistet werden:

* Leider funktioniert der Ansatz nur für Windows-Betriebssysteme, eine Offline-Lösung für Mac-Betriebssysteme ist zum derzeitigen Stand nicht verfügbar
* Automatische Feature-Elimination: Slider, um beim Modellierungen eine Variablenauswahl via Recursive Feature Elimination vorzunehmen. Entsprechend müssten Validierungsoptionen ähnlich zur Hyperparameterauswahl zur Verfügung gestellt werden.
* Datenbank-Überschreibung: Möglichkeit, die prognostizierten Werte im Prediction Tool direkt in eine Datenbank zu schreiben.

Die Unterstützung von Kollaboratoren bei Erstellung dieser erweiterten Funktionalitäten ist ausdrücklich erwünscht. Weitere Funktionalitäts-Vorschläge können auf der zugehörigen Github-Page getätigt werden.

# Abschließende Hinweise:

* Das Dashboard enthält ansprechende Visualisierungen, die auf dem Highcharts-Package beruhen. Eine kommerzielle Nutzung dieser Library erfordert den Kauf einer Lizenz unter dem [folgenden Link](https://shop.highsoft.com/).
* Trotz sorgfältiger Prüfung nach Bugs und ungewollten Verhalten besteht keine Garantie auf Fehlerfreiheit. Zur Bereinigung von Fehlerquellen wird es unter der Plattform Github veröffentlicht und kontinuierlich optimiert.