15.12.20

Einführung in Data Science und maschinelles Lernen mit R

Support Vektor Maschinen

OPENCAMPUS.sh

- **Wiederholung**
- **Definition der Support Vektor Maschine (SVM)**
- **Kreuzvalidierung**
- **Implementierung von SVMs in R**
- **Modellgütekriterien**
- **Einstieg zu neuronalen Netze**
- **Integration von Python Code in RStudio**

BEISPIEL OVERFITTING

```
# Model Prediction Quality for the Training Data Using th
rbind(mape(house_pricing_train$price, predict(mod1)),
      mape(house_pricing_train$price, predict(mod2)),
      mape(house_pricing_train$price, predict(mod3)),
      mape(house_pricing_train$price, predict(mod4)),
      mape(house_pricing_train$price, predict(mod5)),
      mape(house_pricing_train$price, predict(mod6)),
      mape(house_pricing_train$price, predict(mod7)))
```

```
          [,1]
[1,] 0.4328406
[2,] 0.4146068
[3,] 0.2446909
[4,] 0.2422199
[5,] 0.2423780
[6,] 0.2190553
[7,] 0.1781694
```

```
# Model Prediction Quality for the (Unknown) Test Data Using the Mean Absc
rbind(mape(house_pricing_test$price, predict(mod1, newdata=house_pricing_t
      mape(house_pricing_test$price, predict(mod2, newdata=house_pricing_t
      mape(house_pricing_test$price, predict(mod3, newdata=house_pricing_t
      mape(house_pricing_test$price, predict(mod4, newdata=house_pricing_t
      mape(house_pricing_test$price, predict(mod5, newdata=house_pricing_t
      mape(house_pricing_test$price, predict(mod6, newdata=house_pricing_t
      mape(house_pricing_test$price, predict(mod7, newdata=house_pricing_t
```

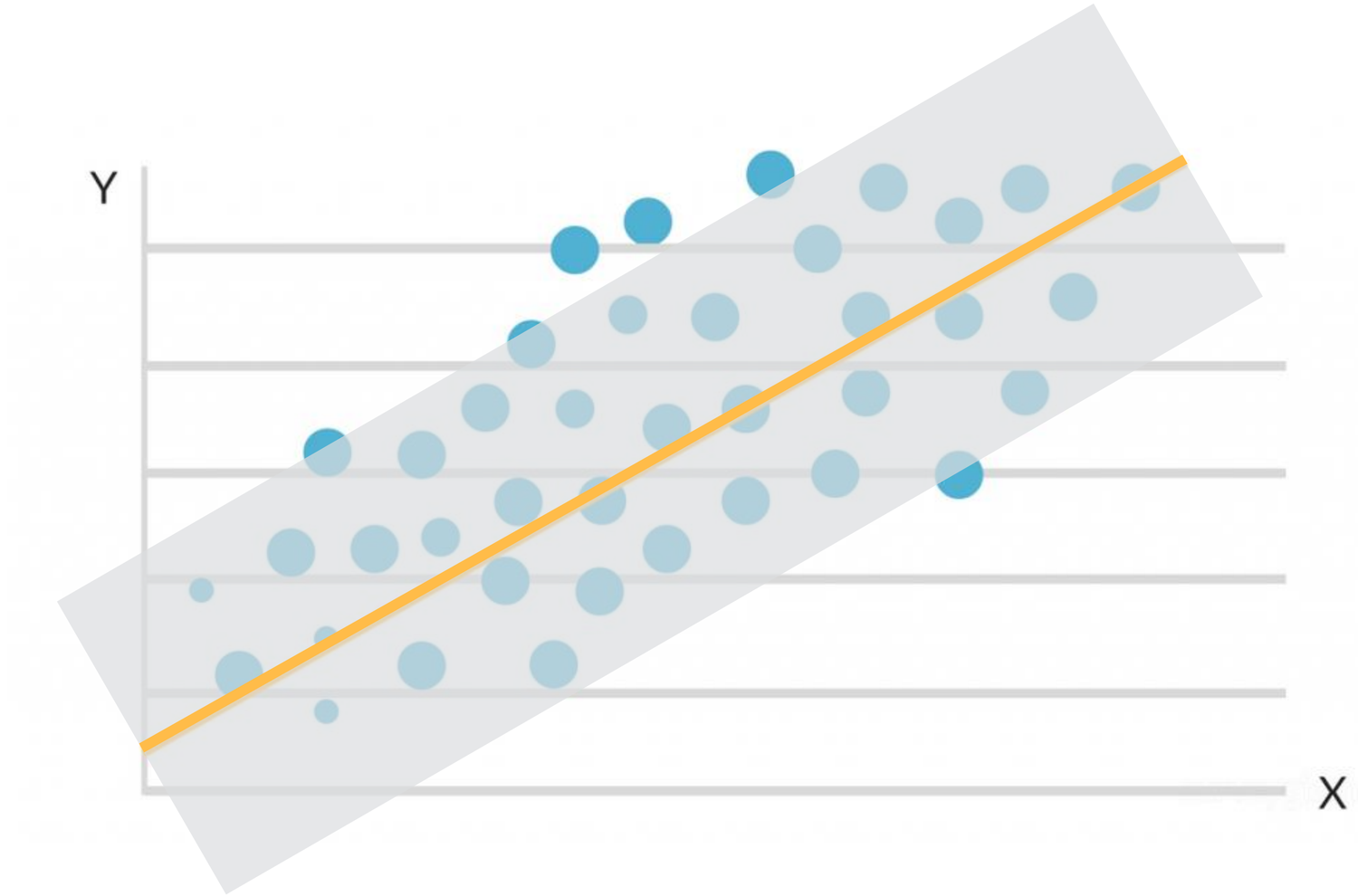
```
          [,1]
[1,] 0.4323069
[2,] 0.4164473
[3,] 0.2473555
[4,] 0.2449223
[5,] 0.2450962
[6,] 0.2230855
[7,] 0.2112316
```

WICHTIGE KONZEPTE

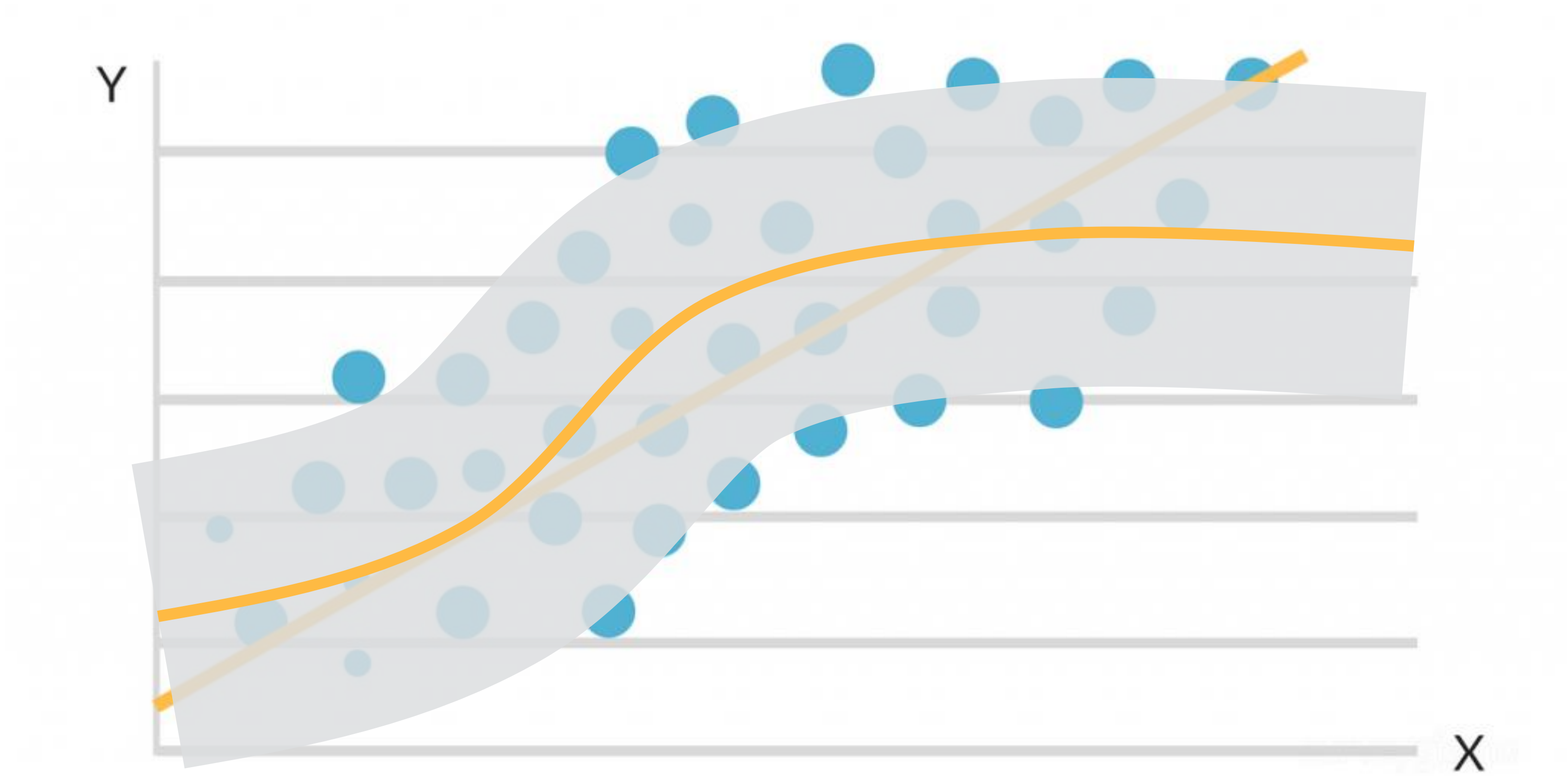
- **Aktivierungsfunktion**
- **Kostenfunktion**
- **Optimierungsfunktion (zur Minimierung der Kostenfunktion)**
- **Lernrate (Eigenschaft der Optimierungsfunktion)**
- **Regularisierung (Bestrafung der Verwendung von Variablen/großen Parametern)**

SUPPORT VECTOR MACHINE (SVM)

[SUPPORT VECTOR REGRESSION (SVR)]



KERNEL TRICK



Blog zur Einführung in SVM:
https://towardsdatascience.com/an-introduction-to-support-vector-regression-svr-a3ebc1672c2

HYPERPARAMETER VON SVM

Modell-Parameter, die keine Gewichte der Modellgleichung sind, werden Hyperparameter genannt.

C-Parameter (auch Cost-Parameter / Soft-Margin-Parameter)

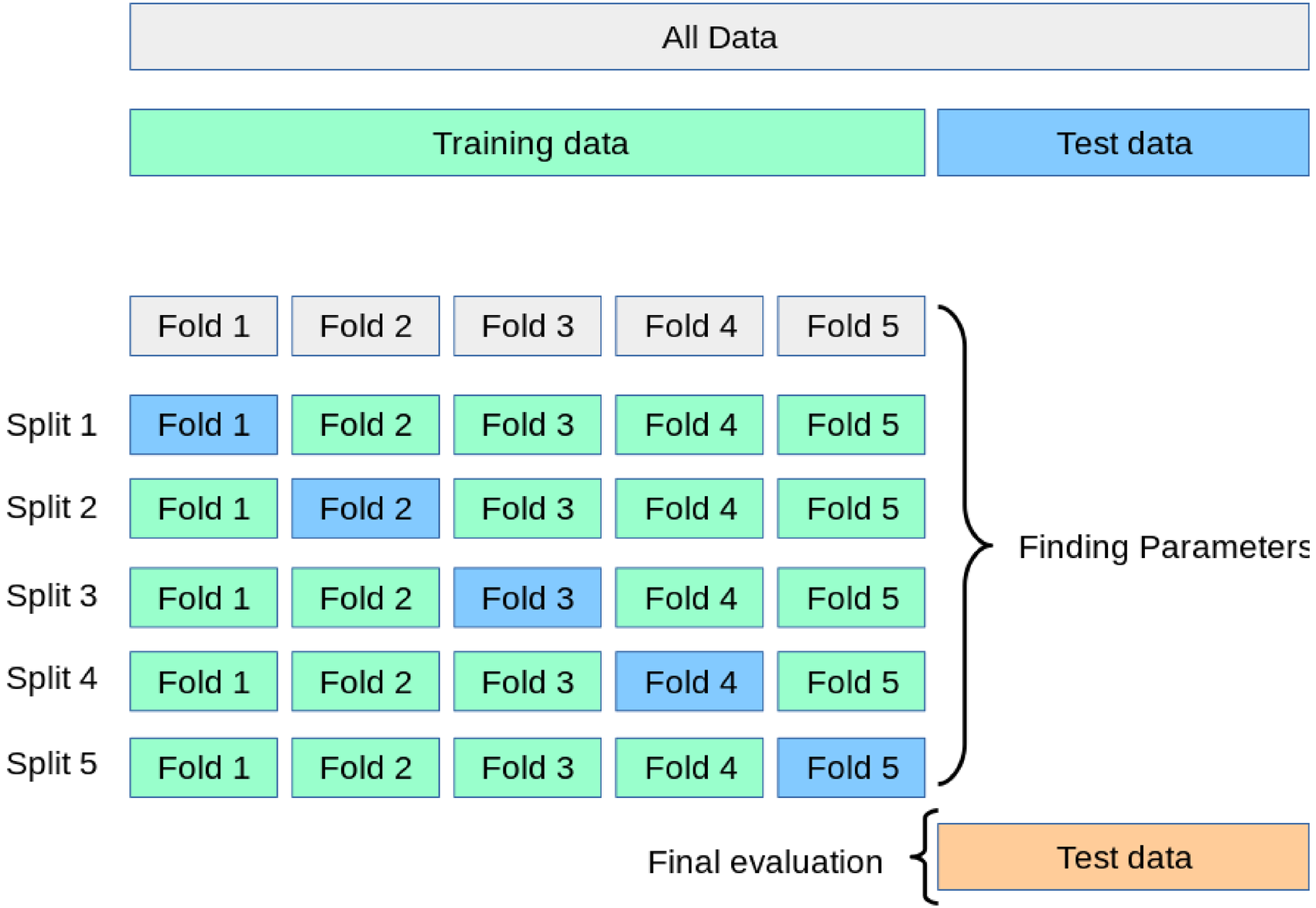
- **Regularisierungsparameter der Kostenfunktion, der die Flexibilität der Grenzen kontrolliert**
- **Kleines C macht die Grenzen flexibel → hohe Varianz / niedriger Bias**
- **Großes C machte die Grenzen starr → niedrige Varianz / hoher Bias**
- **Entspricht $1/\lambda$**

Kernel-Parameter (Gaussian Kernel)

- **Kontrolle der Flexibilität der Kernel-Funktion**
- **Großes Gamma erlaubt hohe Flexibilität → hohe Varianz / niedriger Bias**
- **Kleines Gamma erlaubt geringe Flexibilität → niedrige Varianz / hoher Bias**

N-FOLD CROSS VALIDATION

- **Zufällige Aufteilung des Datensatzes in *n* Gruppen**
- **Schätzung von *n* Modellen wobei jede Gruppe einmal als Testdatensatz genutzt wird und der Rest jeweils immer als Trainingsdatensatz**
- **Die Güte eine Modells (mit fixierten Hyperparametern) wird anhand des Mittelwerts der *n* Schätzungen für die Testdatensätze beurteilt**



Cross-validation: Evaluating estimator performance. (n.d.). Retrieved December 15, 2020, from Scikit-learn 0.23.2 documentation website: https://scikit-learn.org/stable/modules/cross_validation.html

SCHÄTZUNG VON SVM

R-Package „e1071“

`svm()`	**Schätzung eines SVM Modells**
`tune()`	**Schätzung mehrerer SVM Modelle mit Optimierung des C- und Gamma-Parameters auf Basis einer Kreuzvalidierung (Cross Validation)**
`predict()`	**Vorhersage auf Basis eines geschätzten Modells**

TRAINIEREN EINER SVM

```
svm(price ~ bathrooms, train_dataset)
```

```
tune(svm, price ~ bedrooms + bathrooms + sqft_living + zipcode,
                  data=train_dataset,
                  ranges = list(epsilon = seq(0.2,1,0.1),
                  cost = 2^(2:3)))
```

GRUPPENARBEIT

- SVM Modell für Eurem Datensatz trainieren.

ZUSAMMENFASSUNG SVM

- **Sehr populärer, weil einfach zu optimierender Lernalgorithmus, der schnell gute Ergebnisse bringt.**
- **Geeignet zur Klassifizierung und Regression.**
- **Im Fall der (Epsilon-) Regression ist anstelle des Gamma-Parameters ein vergleichbarer Epsilon-Parameter zu setzen.**

KRITERIEN ZUR MODELLGÜTE

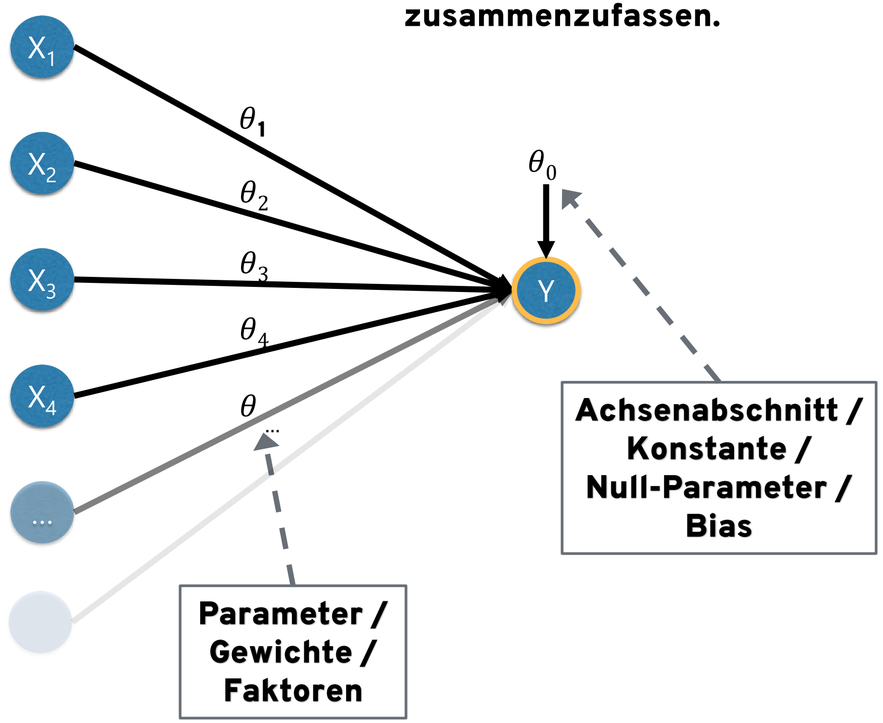
- errors: forecast - actual
- mae: mean(abs(errors))
- mape: mean(abs(errors/actual))
- mse: mean(errors^2)
- rmse: sqrt(mean(errors^2))
- rse: sum(errors^2) / sum(actual-mean(actual))
- r^2 = 1 – rse

Zusätzliches Video (3 Minuten) mit Erklärung und Darstellung der Kriterien:
https://www.coursera.org/lecture/machine-learning-with-python/evaluation-metrics-in-regression-models-5SxtZ

ZUSAMMENFASSUNG LINEARE REGRESSION

Input Layer
Elemente sind die Input-Variablen; auch genannt: Input-Features oder Input-Dimensionen.

Output Layer
Nutzt eine „Aktivierungsfunktion“ (hier lineare Funktion) um die Werte der eingehenden Schicht mit Hilfe der Parameter θ zusammenzufassen.



- **Ziel ist es, anhand des Trainingsdatensatzes die Parameter θ der Aktivierungsfunktion für eine bestmögliche Vorhersage des Testdatensatzes zu optimieren.**

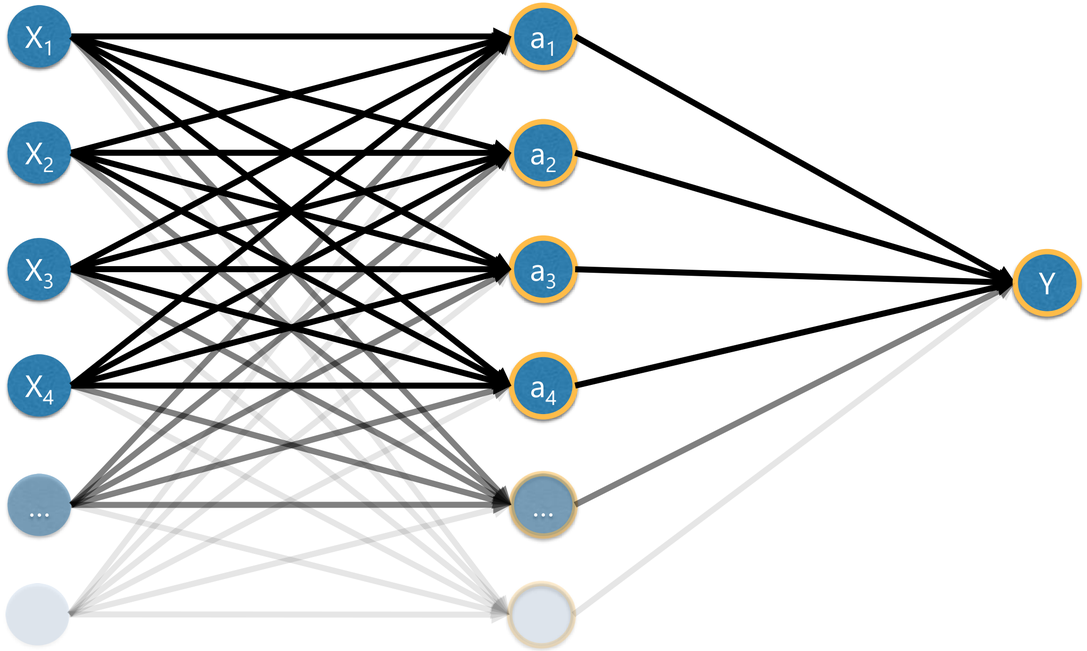
- **Die Verwendung der Gradient Descent Optimierung mit Regularisierung erlaubt, alle Variablen in das Modell eingehen zu lassen und über einen einzelnen Regularisierungsparameter (oder auch Shrinkage-Parameter) den Umfang des Einsatzes der Variablen zu kontrollieren, um so das zur Vorhersage optimale Modell bestimmen zu können.**

NEURONALE NETZE

Input Layer
Besteht aus Input-Variablen / Features / Dimensionen

Hidden Layer
Nutzt eine gegebene „Aktivierungsfunktion“ um die Werte der eingehenden Schicht zusammenzufassen.

Output Layer
Nutzt eine gegebene „Aktivierungsfunktion“ um die Werte der eingehenden Schicht zusammenzufassen.



R VS. PYTHON

Statistische Verfahren außerhalb des ML

Mathematik/Statistik und Disziplinen mit Anwendungsbereichen von Statistik (Ökonometrie, Psychometrie, Biometrie, ...)

Statistische Verfahren zum ML

Angewandte Informatik und freie Wirtschaft mit Anwendungsbereichen von ML

USING PYTHON IN RSTUDIO

R Markdown Python Engine: library (reticulate)

````
```{r}
# Import Libraries
library(reticulate)


# Importing Data
data <- mpg

```


```{python}
year = r.data['year']

```



```{r}
table(py$year)

```
````

See: https://rstudio.github.io/reticulate/articles/r_markdown.html

DATENAUFBEREITUNG FÜR TENSORFLOW

```
### Vorbereitung der Umgebung ###
### Funktionsdefinitionen ###
### Datenimport ###
### Datenaufbereitung ###

```{r}
# Rekodierung von kategoriellen Variablen (zu Dummy-Variablen)
dummy_list <- c("view", "condition")
house_pricing_dummy = dummy_cols(house_pricing, dummy_list)

# Definition von Variablenlisten für die Dummies, um das Arbeiten mit diesen zu erleichtern
condition_dummies = c('condition_1', 'condition_2', 'condition_3', 'condition_4', 'condition_5')
view_dummies = c('view_0', 'view_1', 'view_2', 'view_3','view_4')


# Standardisierung aller Feature Variablen und der Label Variable
norm_list <- c("price", "sqft_lot", "bathrooms", "grade", "waterfront", view_dummies, condition_dumm
norm_values_list <- get.norm_values(house_pricing_dummy, norm_list)     # Berechnung der Mittelwerte
house_pricing_norm <- norm_cols(house_pricing_dummy, norm_values_list) # Standardisierung der Variab
```


Definition der Feaure-Variablen und der Label-Variable
Definition von Trainings- und Testdatensatz
```

RStudio v1.4.1087-9 Preview - Release Notes

This is a preview release of RStudio 1.4, a major new release of RStudio. Highlights include:

- A new **visual editor** for R Markdown documents
- Improved support for Python, including an **environment pane for Python** and visualization of Python objects
- Workbench productivity improvements, including a **command palette** and **rainbow parentheses**
- A more configurable workspace with **additional source columns and improved accessibility**
- Support for **SAML** and **OpenID** authentication, and experimental support for **VS Code** sessions, in RStudio Server Pro
- Dozens of small improvements and bugfixes

See the v1.4.1087-9 Release Notes for full details on all of the changes in this release.

https://rstudio.com/products/rstudio/download/preview/

RStudio v1.4.1087-9 Preview - Release Notes

This is a preview release of RStudio 1.4, a major new release of RStudio. Highlights include:

- A new **visual editor** for R Markdown documents
- Improved support for Python, including an **environment pane for Python** and visualization of Python objects
- Workbench productivity improvements, including a **command palette** and **rainbow parentheses**
- A more configurable workspace with **additional source columns and improved accessibility**
- Support for **SAML** and **OpenID** authentication, and experimental support for **VS Code** sessions, in RStudio Server Pro
- Dozens of small improvements and bugfixes

See the v1.4.1087-9 Release Notes for full details on all of the changes in this release.

Desktop Version

Installers	Size	Date	SHA-256
RStudio 1.4.1087 - Ubuntu 18/Debian 10 (64-bit)	121.19 MB	2020-12-08	3c509bae
RStudio 1.4.1087 - Fedora 28/Red Hat 8 (64-bit)	137.99 MB	2020-12-08	b81a1617
RStudio 1.4.1087 - Debian 9 (64-bit)	121.55 MB	2020-12-08	317d2c38
RStudio 1.4.1087 - macOS 10.13+ (64-bit)	152.78 MB	2020-12-08	5be4baf7
RStudio 1.4.1087 - OpenSUSE 15 (64-bit)	121.89 MB	2020-12-08	27710f3f
RStudio 1.4.1087 - Fedora 19/Red Hat 7 (64-bit)	138.01 MB	2020-12-08	5b720e9a
RStudio 1.4.1087 - Windows 10/8/7 (64-bit)	156.52 MB	2020-12-08	aefbf096
RStudio 1.4.1087 - Ubuntu 16 (64-bit)	118.57 MB	2020-12-08	ff9e78ca

AUFGABEN

- Modellvariablen weiter optimieren!
- SVM für Euren Datensatz trainieren und eine Vorhersage für den 05.06.2019 erstellen.
- Schaut Euch sehr genau dieses Video zur Einführung in Neuronale Netze an (12 Minuten): https://www.youtube.com/watch?v=GvQwE2OhL8I
- Installation von Python (bzw. Miniconda) und den zugehörigen Paketen für R
- Installation von RStudio Version 1.4