## Flugpreisvorhersage

# Kaufen oder Warten?

Data Mining Semesterprojekt WS 2021/22

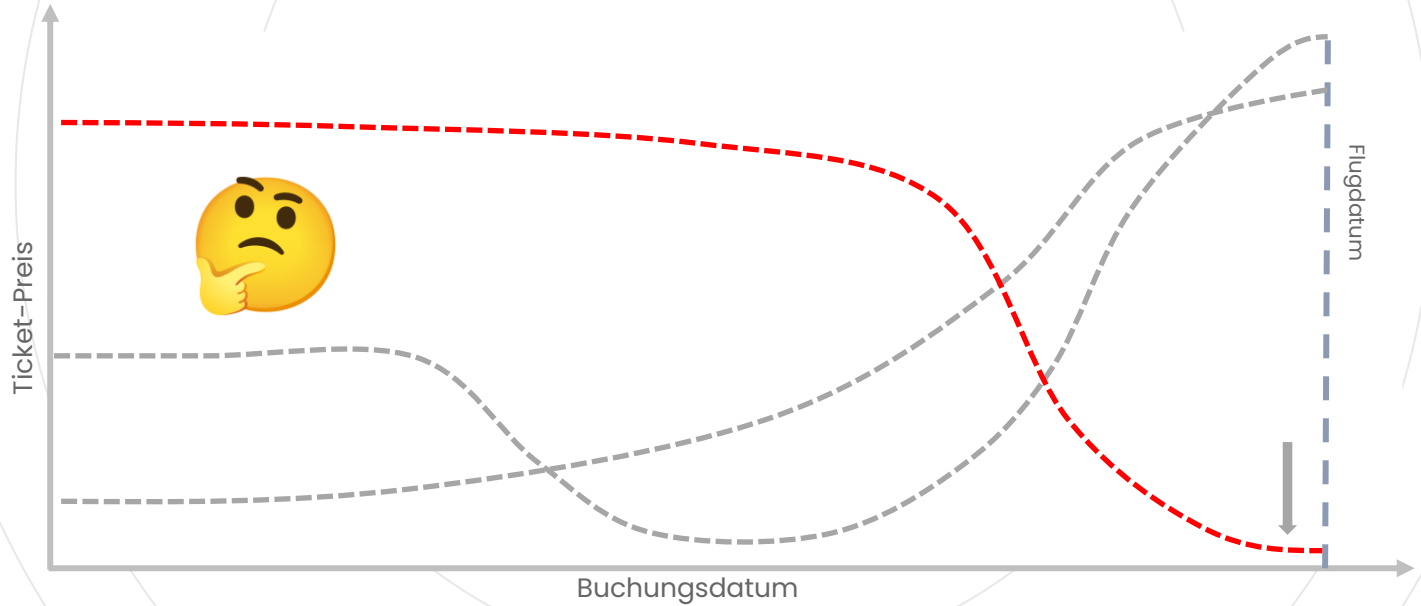
von Max Grundmann - s0559326

Hochschule für Technik und Wirtschaft Berlin

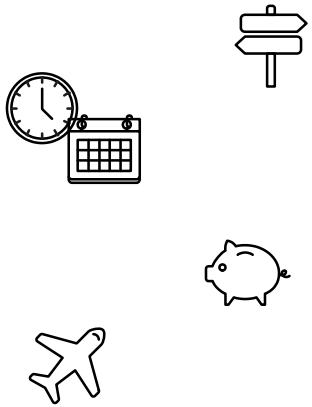
# Agenda

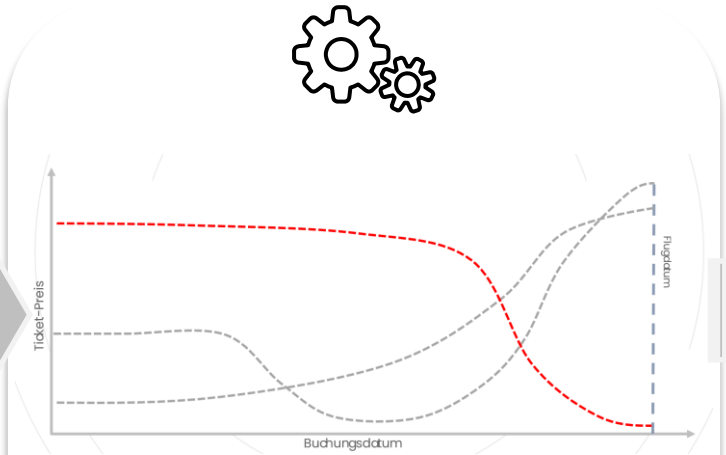
1. Problembeschreibung
2. Modell Auswahl & Training
3. Feature Engineering
4. Ergebnisse

# Sollte ich sofort kaufen oder warten?

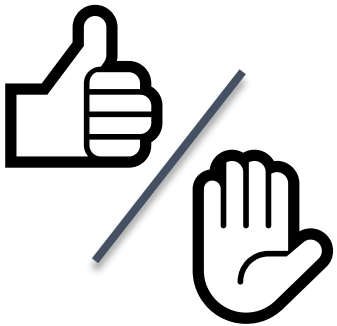


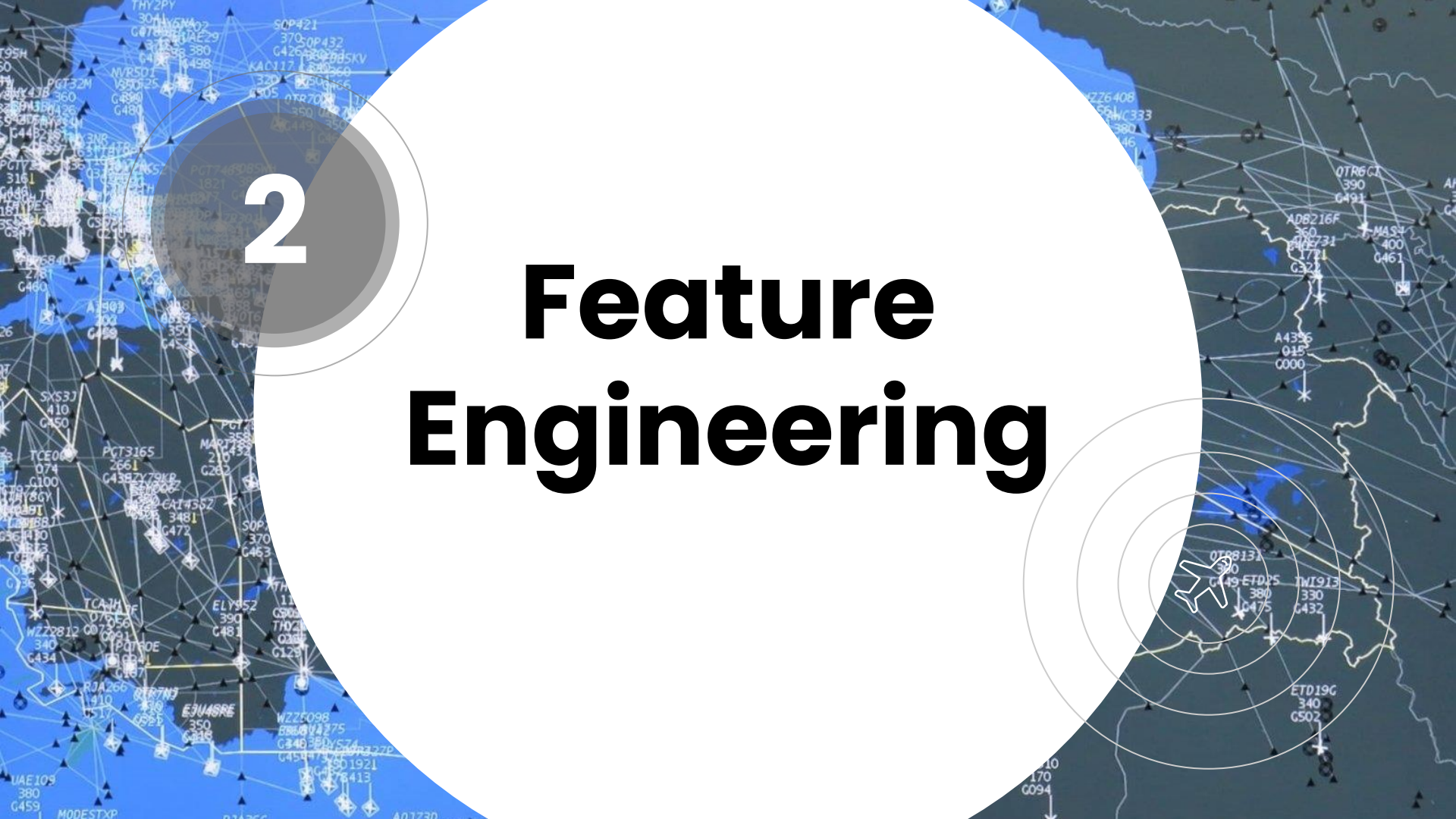
## Input



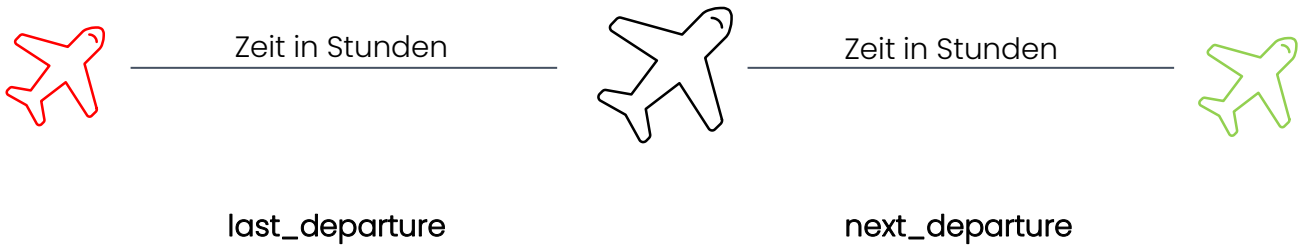



## Output




2
Feature
Engineering

# **Alternative Flüge** auf der gleichen Strecke



# **Alternative Preise** für angrenzende Flüge

price_of_previous_flight

Unter Berücksichtigung des Zeitpunkts der Abfrage

price_of_next_flight

# **Preishistorie** für *n* Schritte

01.04.2021 11:00 Uhr
**180€**

01.04.2021 23:00 Uhr
**190€**

02.04.2021 11:00 Uhr
**210€**

02.04.2021 23:00 Uhr
**240€**

03.04.2021 11:00 Uhr
**260€**

Letzten *n* Preise.
Finaler Datensatz erfasst
bis zu 15 Schritte.

**Preishistorie 0-n**

---

**PriceChange**

Prozentuale Änderung zum
letzten Preis

# Vorherige Abfragen

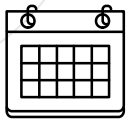
03.04.2021 11:00 Uhr
x 23 Anfragen
für Flug XXX.FR 173

previous_requests

Ist letzte Abfrage
vor dem Flug?

is_last_request

# Tage bis...



Zeit in Tagen

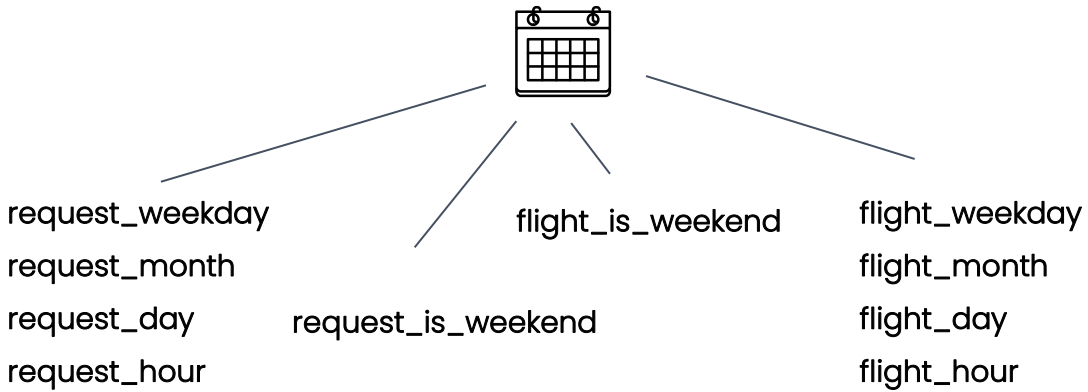
Zeit in Tagen

days_remaining

Days_Untill_Event

```
feiertage = {
    '2019-06-09':'Pfingstsonntag',
    '2019-06-10':'Pfingstmontag',
    '2019-06-20':'Fronleichnam',
    '2019-06-20':'Schulferien Beginn',
    '2019-08-02':'Schulferien Ende',
    '2019-08-26':'Summer Bank Holidays',
    '2019-07-15':'School Summer Holidays Beginn',
    '2019-09-06':'School Summer Holidays End'}
```

# Zerlegung der Datumsfelder



# Abschließend...

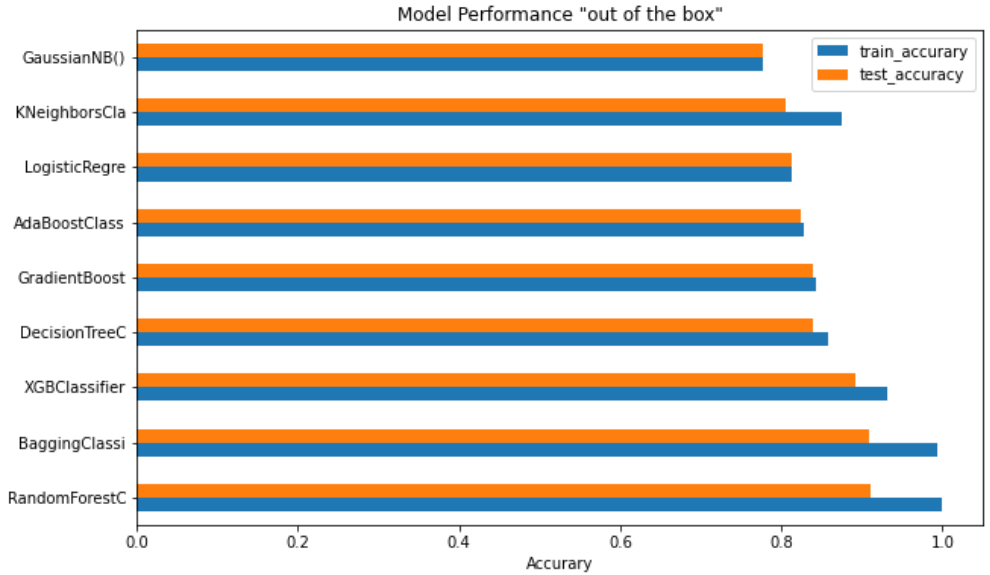
- Datentypen anpassen
- OneHotEncoding für die Routenbezeichnung
- Encoding zyklischer Features Winkel-Abstand als Sinus- und Cosinus Repräsentation
- Skalierung mit MinMaxScaler()
- Entfernung von IDs und Datumsspalten

Übrig bleiben **48 Spalten**

# 3 Model Selection & Training

# Vorauswahl
## „out of the box“

Training der Classifier „out of the box“
und ohne Crossvalidation.



# Vorauswahl mit Crossvalidation cv=5

Anwendung von Crossvalidation auf die vier vielversprechendsten Verfahren

- **Random Forest:** mean Accuracy = 0.906
- **Bagging:** mean Accuracy = 0.901
- **XGBoost:** mean Accuracy = 0.890
- **Decison Tree:** mean Accuracy = 0.842

# Vorauswahl mit Crossvalidation cv=5

Anwendung von Crossvalidation auf die vier vielversprechendsten Verfahren

- **Random Forest:** mean Accuracy = 0.906
- **Bagging:** mean Accuracy = 0.901
- **XGBoost:** mean Accuracy = 0.890
- **Decison Tree:** mean Accuracy = 0.842

# Randomized Search CV

Zufallsbasierte Suche der besten Parameter mit 30 Ausführungen und jeweils 5fold CV

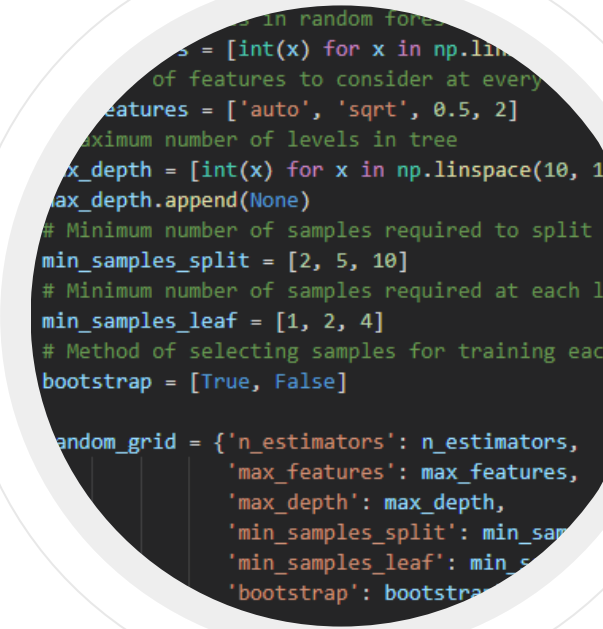
Ausführung auf Google Colab zwecks besserer Performance

**Ergebnis:** grobe Richtung für weiteres Finetuning

```
{'n_estimators': 229, 'min_samples_split': 10,
'min_samples_leaf': 2, 'max_features': 0.5, 'max_depth':
90, 'bootstrap': False}
```

# Grid Search

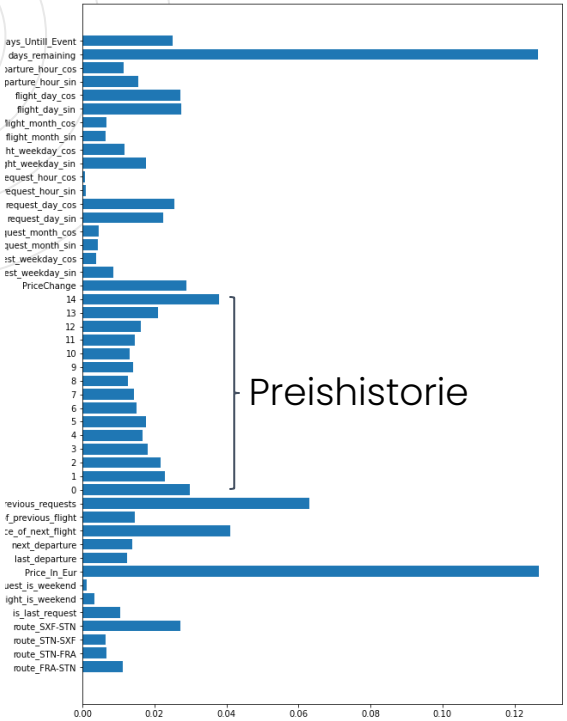
Weitere Eingrenzung der Parameter.




## Finale Hyperparameter:

```
{'bootstrap': False,
'max_depth': 70,
'max_features': 0.75,
'min_samples_leaf': 1,
'min_samples_split': 10,
'n_estimators': 300}
```

# Feature Importance

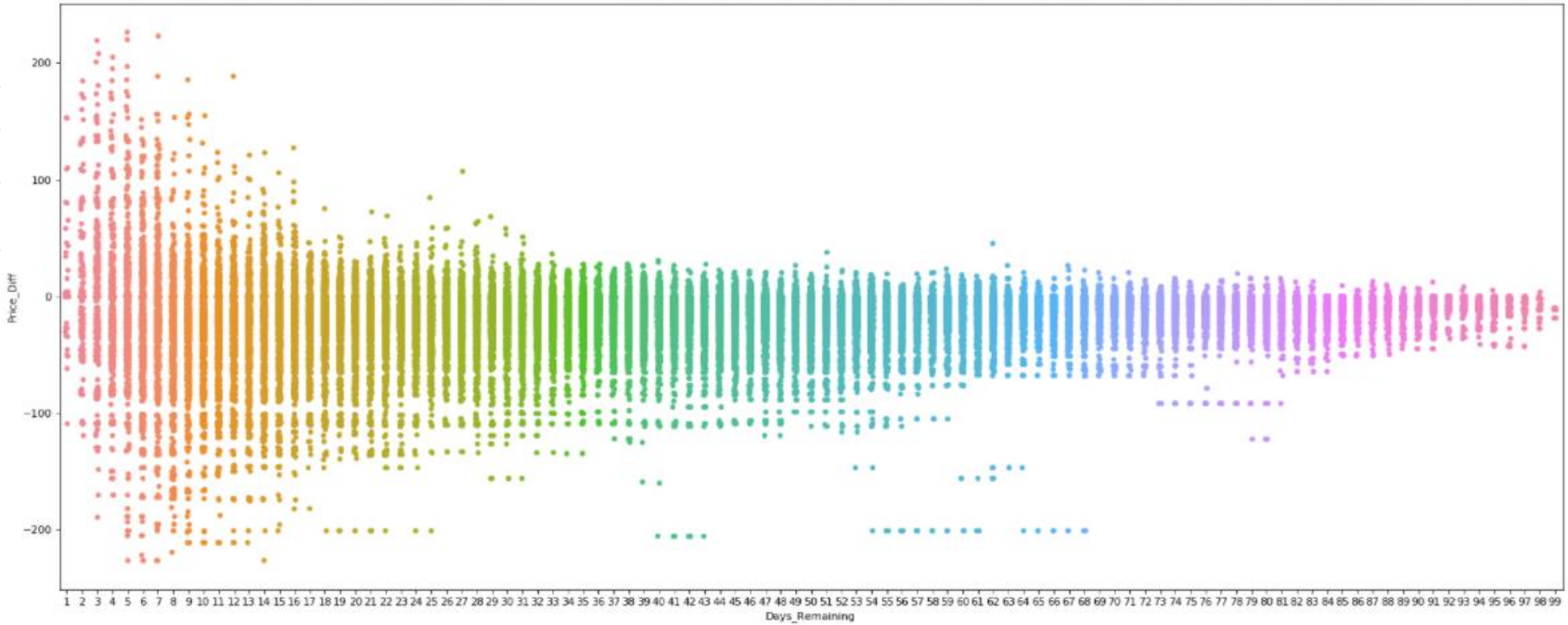



## Top 5 Features

| | |
|---|---|
| Price_In_Eur | 12,7 % |
| days_remainig | 12,6 % |
| previous_requests | 6,3 % |
| price_of_next_flight | 4,1 % |
| (last Price) | 3,7 % |

Die „unwichtigsten“ 9 Features können ohne Accuracy-Verlust entfernt werden.

# Preisdifferenz im zeitlichen Verlauf



# 2. Verfahren: **Neural Network**


Experimentelles Vorgehen

- Testen verschiedener Netzwerkarchitekturen
- Testen verschiedener Optimizer
- Testen verschiedener Batch Größen
- Testen verschiedener Epoch Laufzeiten

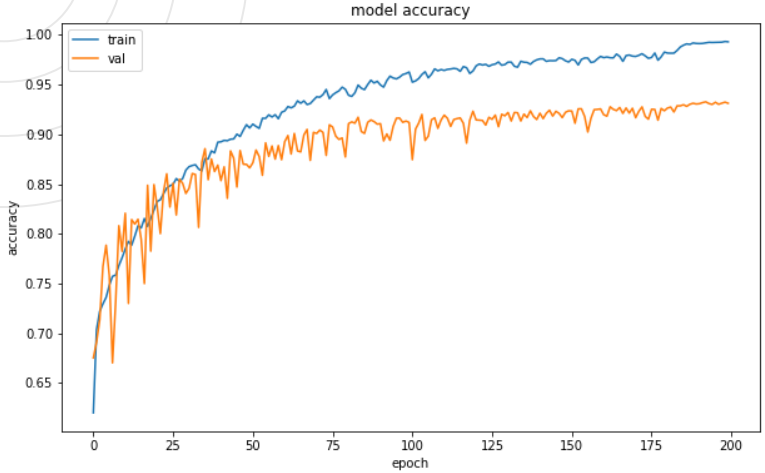
Ebenfalls auf Colab GPU Instanz trainiert.

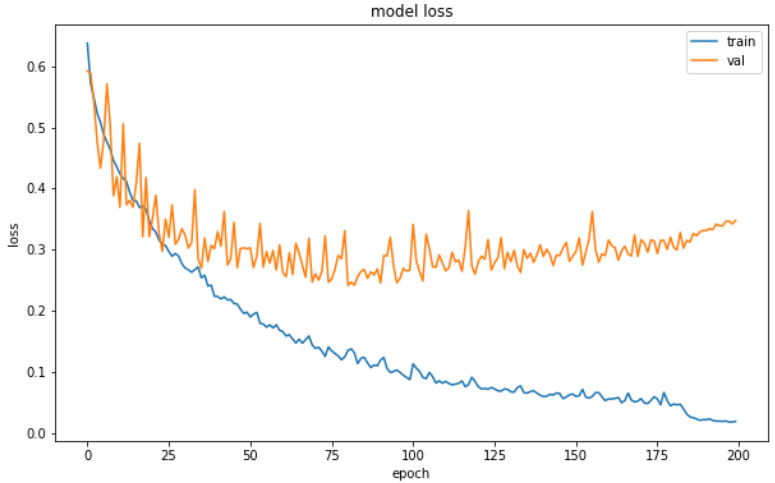
# 2. Verfahren: **Neural Network**

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_6 (Dense)             (None, 512)               24576

 dense_7 (Dense)             (None, 1024)              525312

 dropout_2 (Dropout)         (None, 1024)              0

 dense_8 (Dense)             (None, 512)               524800

 dense_9 (Dense)             (None, 128)               65664

 dropout_3 (Dropout)         (None, 128)               0

 dense_10 (Dense)            (None, 16)                2064

 dense_11 (Dense)            (None, 1)                 17

=================================================================
Total params: 1,142,433
Trainable params: 1,142,433
Non-trainable params: 0
_________________________________________________________________
```

```
optimizer= ‚adam‘
loss = ‚binary_crossentropy‘

batch_size = 4096
epochs = 200
```

# Loss & Accuracy



# 4

# Ergebnisse

# Accuracy

## Random Forest

Test Accuracy: **0.9301**

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.94 | 0.97 | 0.96 | 12981 |
| 1 | 0.88 | 0.80 | 0.84 | 3744 |
| accuracy | | | 0.93 | 16725 |
| macro avg | 0.91 | 0.88 | 0.90 | 16725 |
| weighted avg | 0.93 | 0.93 | 0.93 | 16725 |

## Neural Network

Test Accuracy: **0.9313**

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.96 | 0.95 | 0.96 | 12981 |
| 1 | 0.84 | 0.86 | 0.85 | 3744 |
| accuracy | | | 0.93 | 16725 |
| macro avg | 0.90 | 0.91 | 0.90 | 16725 |
| weighted avg | 0.93 | 0.93 | 0.93 | 16725 |

# Accuracy

## Random Forest

Test Accuracy: **0.9301**

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.94 | 0.97 | 0.96 | 12981 |
| 1 | 0.88 | 0.80 | 0.84 | 3744 |
| accuracy | | | 0.93 | 16725 |
| macro avg | 0.91 | 0.88 | 0.90 | 16725 |
| weighted avg | 0.93 | 0.93 | 0.93 | 16725 |

## Neural Network

Test Accuracy: **0.9313**

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.96 | 0.95 | 0.96 | 12981 |
| 1 | 0.84 | 0.86 | 0.85 | 3744 |
| accuracy | | | 0.93 | 16725 |
| macro avg | 0.90 | 0.91 | 0.90 | 16725 |
| weighted avg | 0.93 | 0.93 | 0.93 | 16725 |

# Accuracy

## Random Forest

Test Accuracy: **0.9301**

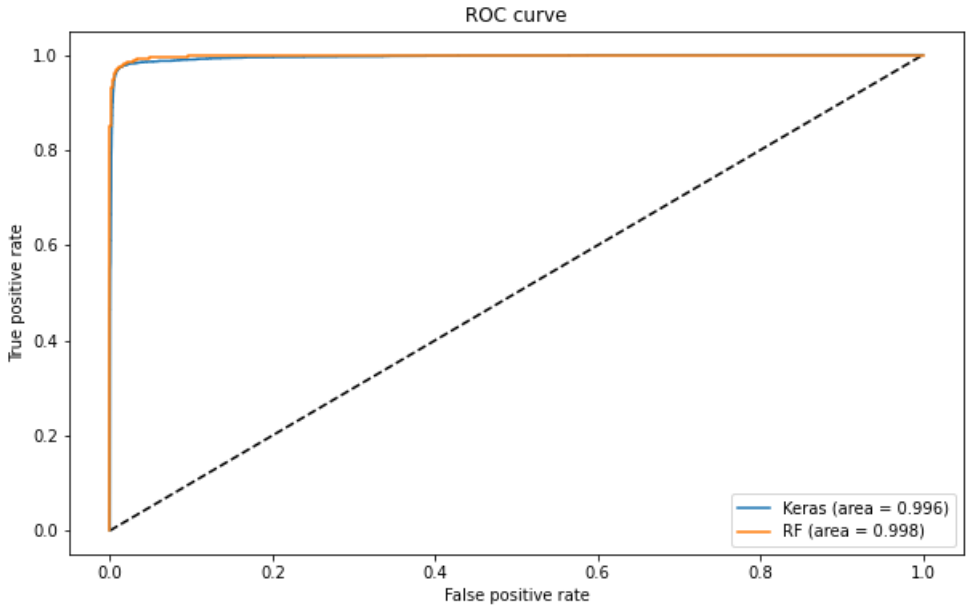
| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.94 | 0.97 | 0.96 | 12981 |
| 1 | 0.88 | 0.80 | 0.84 | 3744 |
| accuracy | | | 0.93 | 16725 |
| macro avg | 0.91 | 0.88 | 0.90 | 16725 |
| weighted avg | 0.93 | 0.93 | 0.93 | 16725 |

## Neural Network

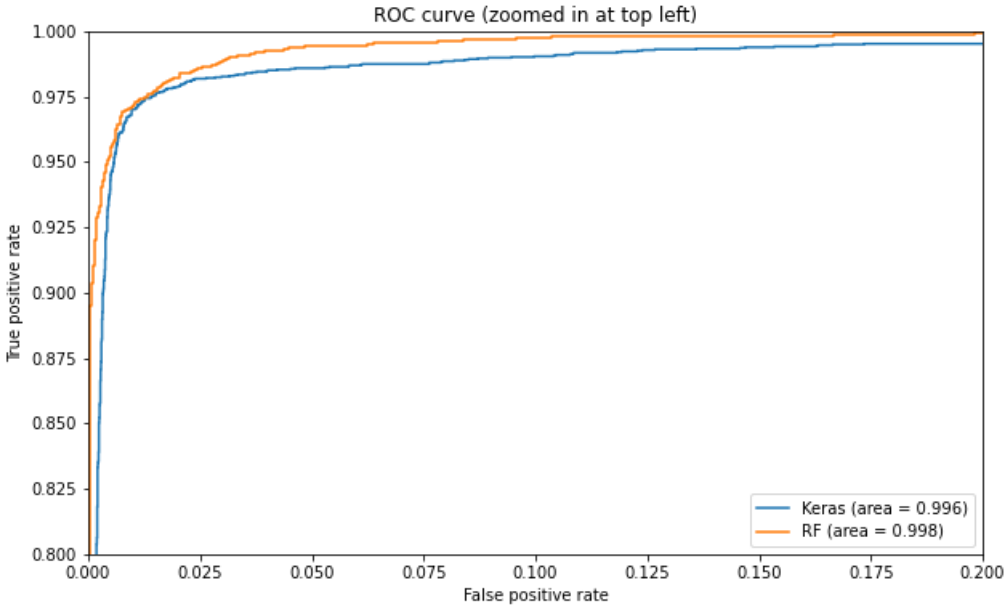
Test Accuracy: **0.9313**

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.96 | 0.95 | 0.96 | 12981 |
| 1 | 0.84 | 0.86 | 0.85 | 3744 |
| accuracy | | | 0.93 | 16725 |
| macro avg | 0.90 | 0.91 | 0.90 | 16725 |
| weighted avg | 0.93 | 0.93 | 0.93 | 16725 |

# AUC-ROC Kurve RF vs. NN



# AUC-ROC Kurve RF vs. NN



# Monetäres Gütemaß

Ground Truth

1.388.860.66€

Random Forest

1.279.392.33€ -109.468,33€ (92,2%)

Neural Network

1.244.365.11€ -144.495,55€ (89,5%)

# Monetäres Gütemaß

Ground Truth

1.388.860.66€

Random Forest

1.279.392.33€ -109.468,33€ (92,2%)

Neural Network

1.244.365.11€ -144.495,55€ (89,5%)

**Aber:** Gütemaß optimiert absolute Ersparnis, nicht prozentuale Ersparnis.

# Mögliches Geschäftsmodell

**Kaufempfehlung**

Empfehlung zum sofortigen Kauf oder Warten.

**Automatisierung**

Benachrichtigung des Nutzers, wenn der Preis am günstigsten ist.

Nutzer zahlt preisabhängige Kommission.
**Kommission auf Basis der Ersparnis**

Erstattung der Differenz, falls der Preis doch steigt.

**Preisgarantie**

# Danke!

**Fragen? Feedback?**

Max Grundmann – s0559326