# Customer Churn Prediction

Lina Groth, Klara Kulinna, Jennifer Tielke, Marc Buddemeier

# Use Case

- Telekommunikationsunternehmen **TelCo America**
- Standort USA
- Anbieter von Telekommunikationsdiensten, einschließlich Telefon- und Internetservices

# Business Understanding

# Ziele

**Welche Ziele hat das Unternehmen mit dem Modell?**

- Identifikation von abwanderungsgefährdeten Kunden
- Ermittlung von möglichen Gründe für Kundenabwanderung

# Ergebnis

**Welches Ergebnis erwünscht sich das Unternehmen aus den Zielen?**

- Reduzierung der Kundenabwanderung
- Erhöhung der Kundenloyalität
- Erhöhung von langfristigen Kundenbeziehungen
- Steigerung der Umsätze durch bessere Kundenbindung

# Prozesse

**Welche Prozesse sollen durch das Modell verbessert werden?**

- *Kundenservice-Prozesse*
  - Verbesserung der Reaktionsfähigkeit bei drohender Kundenabwanderung
  - Entwicklung von Frühwarnsystemen zur frühzeitigen Erkennung von Kundenabwanderung
- *Marketing- und Verkaufsstrategien*
  - Entwicklung gezielter Marketingkampagnen für abwanderungsgefährdete Kunden

# Klassifikation

- Binäre Klassifikation von Kunden anhand von Kundendaten

| Wahrheit / Vorhersage | Churn | Retained |
|---|---|---|
| Churn | 0 (True Positive) | -1 (False Positive) |
| Retained | -5 (False Negative) | 0 (True Negative) |

richtige Vorhersage
(keine Kosten)

Falsche Vorhersage
(Kunde wandert ab)

Falsche Vorhersage
(Kunde bleibt doch)

# Data Understanding

# Vorhandene Informationen

## Features

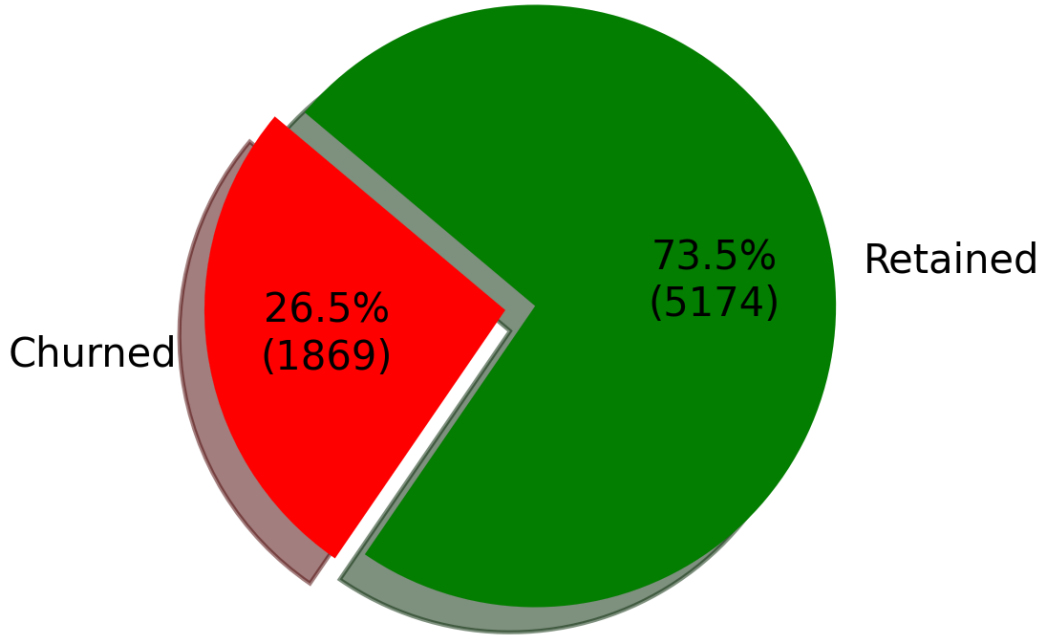
- *Demografische Daten*: Geschlecht, Alter, Verheiratet, Anzahl der Kinder
- *Geographische Daten*: Land, Bundesstaat, Stadt, Postleitzahl, Koordinaten und Bevölkerungsanzahl der Städte
- *Gebuchte Dienste*: monatliche Downloadraten, Vertragsdaten, monatliche Gebühren, ...
- *Kundenstatus*: Zufriedensheitsscore

## Label

- Churn (No/Yes)

# Label

- 7.043 Datenpunkte
- Klasse "Retained" ist überrepräsentiert



# Data Preparation

# Spalten kodieren

## Spalten entfernen

- **Technische Identifier**
  - Keine inhaltliche Bedeutung
  - Customer ID, Location ID
- **Konstante Spalten**
  - Wert ist für jede Entität gleich
  - Country (nur USA), State (nur California), Quarter, Count

## Strings kodieren

- Kodierung als True/False, falls es nur zwei unique Values gibt
- Kodierung mittels OneHot-Encoding, falls es viele Werte gibt

# Spalten kodieren

```
for col in df.columns: # Iterate through each column
    # If the column has only one unique value or is technical ID, drop it because it cannot be used for prediction
    if df[col].nunique() == 1 or col in coloumns_to_drop:
        df_encoded = df_encoded.drop(columns=[col])
    # If the column is a string column, encode it
    elif col in string_columns:
        if df[col].nunique() == 2: # If the column has only two unique values, use label encoding
            df_encoded[col] = label_encoder.fit_transform(df[col])
        else: # If the column has more than two unique values, use one-hot encoding
            df_encoded = pd.get_dummies(df_encoded, columns=[col], prefix=[col])
    else:
        continue
```

# Data Understanding

# Korrelationen zum "Churn"-Label

| Feature | Beträge der Korrelation |
|---|---|
| Satisfaction Score | 0,755 |
| Contract_Month-to-Month | 0,448 |
| Tenure in Months | 0,352 |
| Contract_Two Year | 0,328 |
| ... | ... |
| City_Los Angeles | 0.000397 |
| City_Ontario | 0.000136 |

- Kundenzufriedenheit und Vertragsdauer zeigen hohe Korrelationen
- Orte korrelieren kaum

# Modeling

# Mögliche Modelle

- Entscheidungsbäume (Bagging & Boosting)
- Naive Bayes
- Logistische Regression
- Support Vector Machines
- k-Nearest Neighbors
- Neuronale Netze

# Vorgehensweise

- Verwendung von SMOTE, um die Klassenungleichgewichte auszugleichen
- ggf. Features skalieren, falls es für das Modell benötigt wird
- Verwendung k-Fold-Cross-Validation, um Modelle zu beurteilen
- Hyper-Parameter-Tuning mittels Grid Search

# Evaluation

# Metriken zur Modellbewertung

- $\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$
- $\text{Precision} = \frac{TP}{TP+FP}$
- $\text{Recall} = \frac{TP}{TP+FN}$
- $\text{F1 Score} = \frac{2}{\frac{1}{\text{Precision}}+\frac{1}{\text{Recall}}}$
- $\text{Kosten} = FP + 5 \cdot FN$

# Praktische Umsetzung

# Bayes-Theorem

- Bedingte Wahrscheinlichkeiten
- Nimmt Featureunabhängigkeit an

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}$$

# Naive Bayes

# Vor- und Nachteile

- Einfachheit
- Hohe Recheneffizienz
- Bei großen Datensätzen hohe erreichbare Genauigkeit

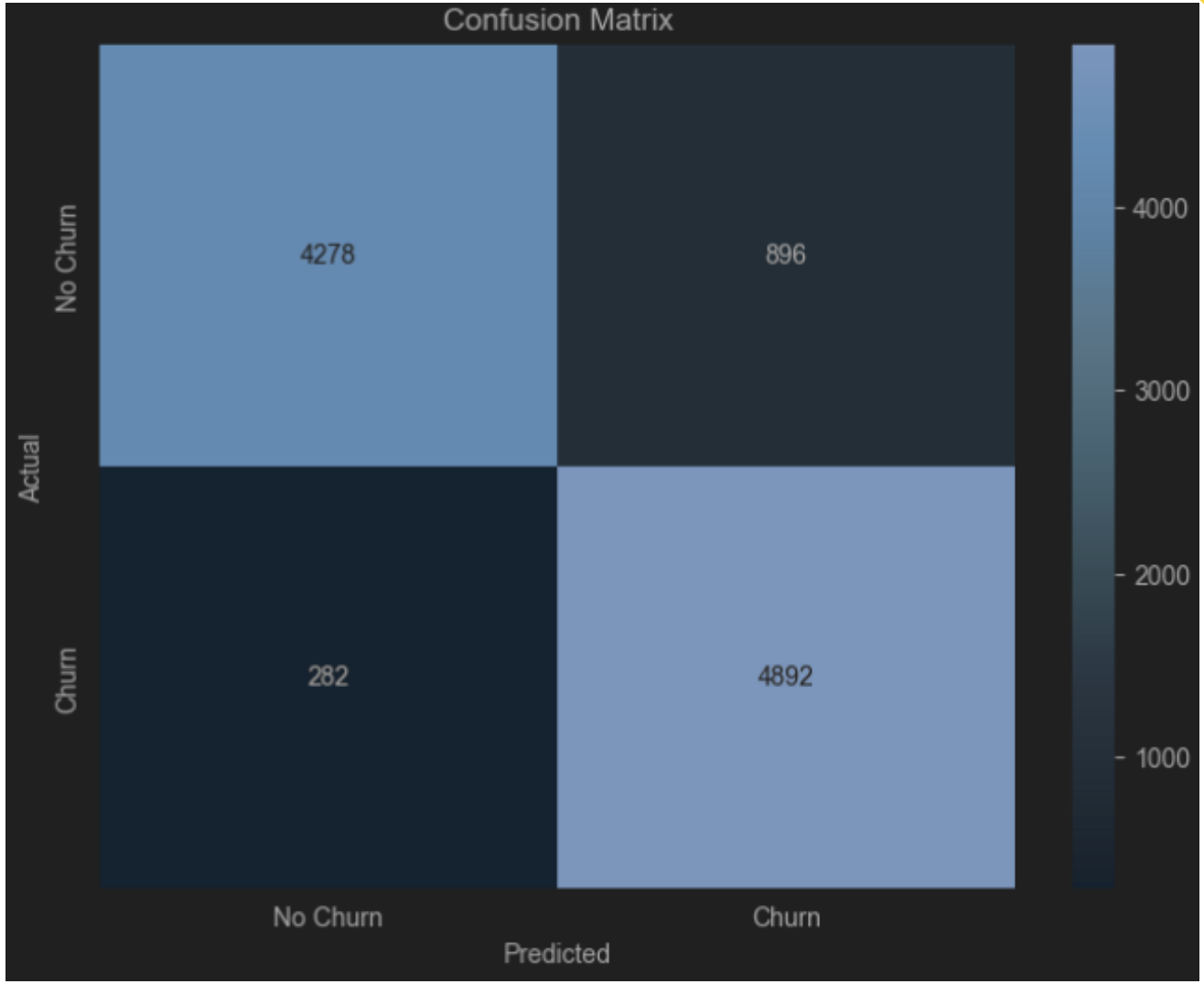
- Unabhängigkeitsannahme
- Keine komplexen Beziehungen modellierbar
- Empfindlichkeit gegenüber seltenen Ereignissen

# Ergebnisse

Mit Gaussian Naive Bayes

Accuracy:

"nur" 89%



# Bagging und Boosting

### RandomForestClassifier

- Kombination von Vorhersagen

  → Verringerung der Varianz

- Viele Entscheidungsbäume aus verschiedenen Stichproben
- Verringert Overfitting

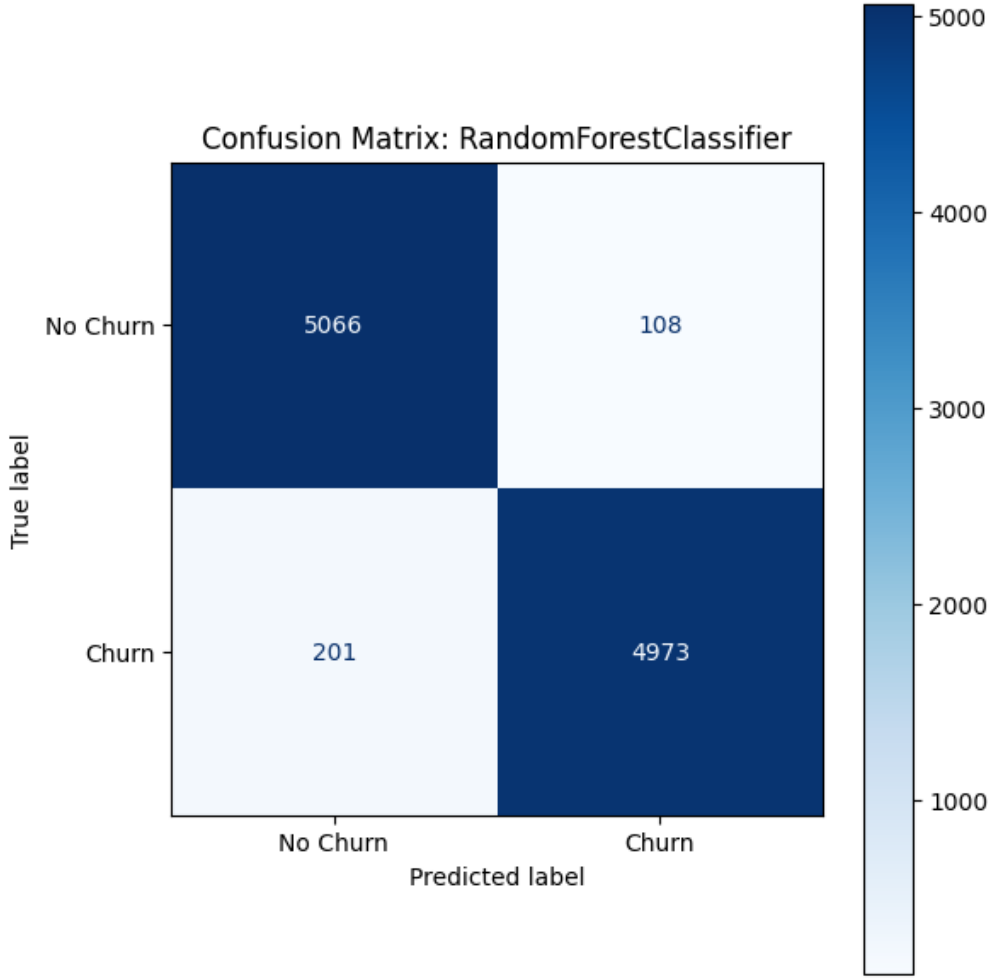
### AdaBoostClassifier

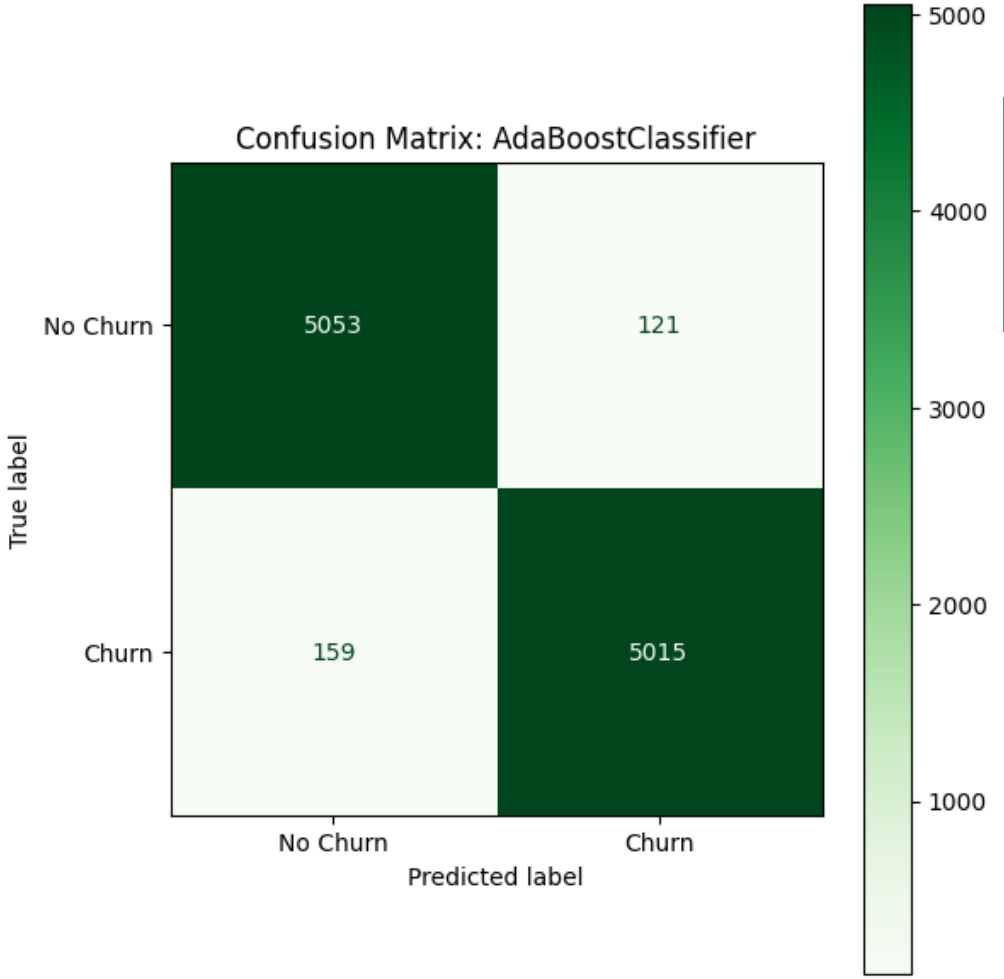
- Kombination vieler einfacher Modelle, um ein starkes Modell zu erstellen
- Schwache Entscheidungsbäume werden trainiert
- Ergebnisse werden neu gewichtet
- Reduzierung des Bias

# Ergebnisse

**Accuracy Boosting:**
96%

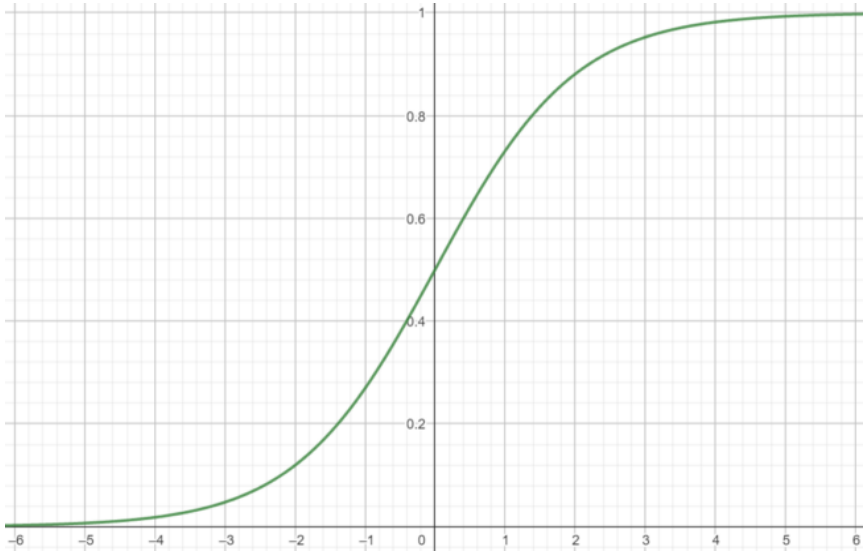
**Accuracy Bagging:**
95%



# Logistische Regression

Logistische Funktion




- Für binäre Ereignisse
- Wahrscheinlichkeitsberechnung für eine Linearkombination von Koeffizienten

$$P(Y = 1|X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n)}}$$

# Maximum-Likelihood-Schätzung

- Zur Identifizierung der β
- Kann über verschiedene Algotithmen durchgeführt werden:
  - Lbfgs: Limited-memory Broyden-Fletcher-Goldfarb-Shanno
  - Liblinear: **Coordinate Descent**-Algorithmus
  - Saga: Stochastic Average Gradient Augmented

```
clf = (
    LogisticRegression(
        max_iter=2000, solver='lbfgs',
        random_state=42))
```

```
param_grid = {
    'max_iter': [5000, 10000],
    'solver': ['lbfgs', 'liblinear'],
    'C': [0.1, 1, 10]
}
```

# Vorteile und Nachteile

- Interpretierbarkeit
- Hohe Recheneffizienz
- Wahrscheinlichkeitsauswertung
  - Wahrscheinlichkeit pro Ereignis sichtbar
  - Macht Grenzfälle sichtbar

- Lineare Entscheidungsgrenze
- Empfindlichkeit gegenüber Ausreißern
- Multikollinearität
- Benötigt ausreichend Daten

# Deployment

# Deployment

- Modell exportieren
- Täglicher automatischer Analyse-Workflow
  - Alle Kunden werden anhand aktueller Daten eingeschätzt
  - Ergebnisse werden in Dashboard bereitgestellt
- Ausblick: Explainable ML
  - Modell soll erklären können, warum es "Churn" vorhersagt, um genau dort gegensteuern zu können

# Überwachung des Modells

- Monatlich werden Predictions gespeichert und mit tatsächlichem Kundenverhalten verglichen
- Tatsächliche Accuracy wird ebenfalls im Dashboard bereitgestellt
- Falls Accuracy unter 90 % fällt, wird das Modell mit neuen Daten automatisch trainiert
- Falls Accuracy nicht wieder 90 % erreicht, automatische Benachrichtung

→ Erneute manuelle Evaluierung der Parameter und Modelle

# Fazit

# Fazit

- Modell hat mit ca. 97,5 % eine sehr hohe Genauigkeit und kann damit dem Kundensupport helfen, Kunden zu identifizieren, die drohen abzuwandern

- Dashboard ermöglicht einfache Verwendung des Modells, um abwanderungsgefährdete Kunden zu identifizieren, um die Gründe dahinter weiter zu analysieren

# Kritische Reflexion

- Nutzer des Modells erhalten keine Erklärung, warum Kunde abwandern möchte
- Festlegung der Kosten für FN und FP war eher willkürlich
- Künstlich erstellter Datensatz, sodass Accuracy mit echten Daten evtl. nicht erreichbar ist
- Ausgewähltes Modell zeigte höchste Accuracy
  - Es gibt unendliche viele Kombinationen für ausgewählte Modelle, Features, etc.
  - Andere Modelle können evtl. bessere Ergebnisse erreichen

# Vielen Dank für Ihre Aufmerksamkeit!