Visualisierung – Wann welche Darstellung?

Exploration anhand des Titanic-Datensatzes

Leitfrage: Welche Visualisierung ist für welche Fragestellung geeignet – und warum?

Datensatz *Quelle*: https://github.com/mwaskom/seaborn-data/blob/master/titanic.csv
Bereitgestellt von: https://seaborn.pydata.org/

Jupyter Notebook von: Devin Rohrer

In [1]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Originaler titanic datensatz
df = sns.load_dataset('titanic')

# Styling
sns.set_theme(style='whitegrid', palette='muted')
plt.rcParams['figure.dpi'] = 110

print(f"Datensatz: {df.shape[0]} Passagiere, {df.shape[1]} Merkmale")
print(f"Überlebensrate gesamt: {df['survived'].mean():.1%}")

# Die ersten 5 Zeilen ausgeben
df.head()
```

```
Datensatz: 891 Passagiere, 15 Merkmale
Überlebensrate gesamt: 38.4%
```

Out[1]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult
0	0	3	male	22.0	1	0	7.2500	S	Third	man	
1	1	1	female	38.0	1	0	71.2833	C	First	woman	
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	
3	1	1	female	35.0	1	0	53.1000	S	First	woman	
4	0	3	male	35.0	0	0	8.0500	S	Third	man	

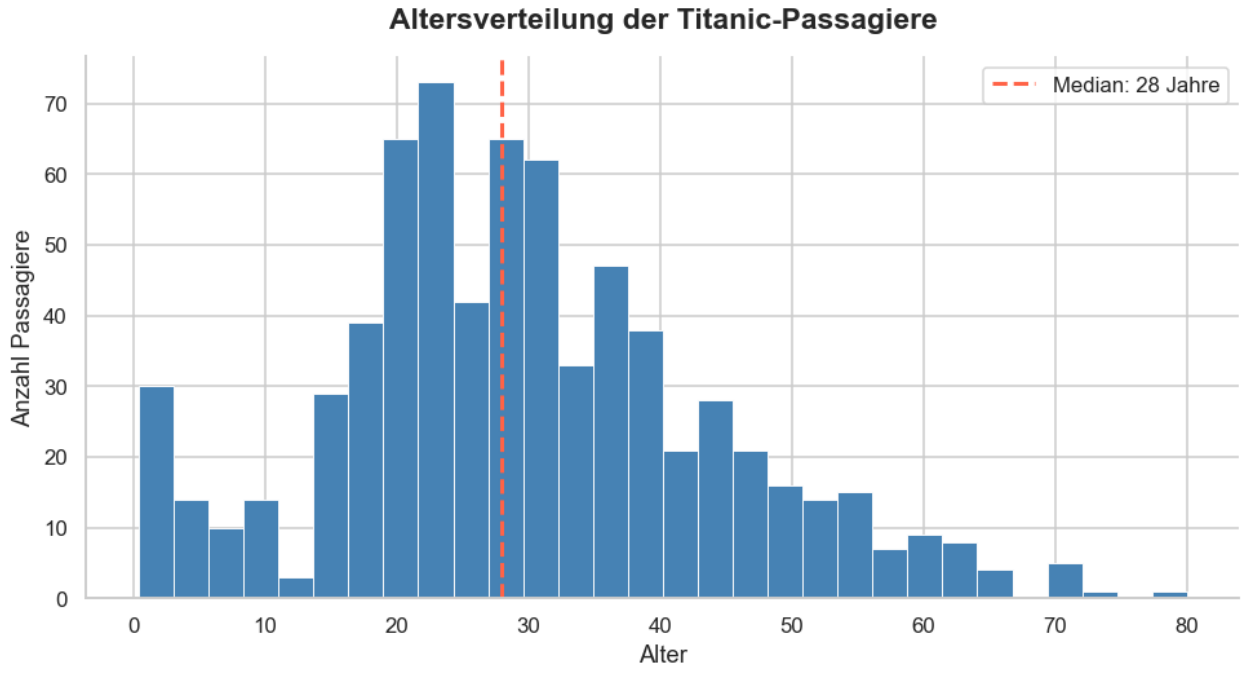
1. Visualisierung: Histogramm

Wann ein Histogramm?

Immer dann, wenn man eine **einzelne numerische Variable** hat und verstehen will, wie die Werte verteilt sind – wo die Masse liegt, ob es Ausreißer gibt, ob die Verteilung symmetrisch oder schief ist.

Frage: Wie ist das Alter der Passagiere verteilt?

In [2]:
```
fig, ax = plt.subplots(figsize=(9, 5))

# Histogramm erstellen:
ax.hist(df['age'], bins=30, color='steelblue', edgecolor='white', linewidth=0.

# Median berechnen und visualisieren:
median_age = df['age'].median()
ax.axvline(median_age, color='tomato', linestyle='--', linewidth=2,
           label=f'Median: {median_age:.0f} Jahre')

ax.set_title('Altersverteilung der Titanic-Passagiere', fontsize=15, fontweigh
ax.set_xlabel('Alter', fontsize=12)
ax.set_ylabel('Anzahl Passagiere', fontsize=12)
ax.legend(fontsize=11)

sns.despine()
plt.tight_layout()
plt.show()

print(f"Median: {df['age'].median():.0f} Jahre | Mittelwert: {df['age'].mean()
print(f"Jüngster: {df['age'].min():.0f} Jahre | Ältester: {df['age'].max():.0f
```




```
Median: 28 Jahre | Mittelwert: 29.7 Jahre
Jüngster: 0 Jahre | Ältester: 80 Jahre
```

Beobachtung: Die meisten Passagiere waren zwischen 20 und 40 Jahre alt. Es gibt einen kleinen Peak bei Kindern und einige ältere Passagiere als Ausreißer.

Wann kein Histogramm? Wenn man Gruppen miteinander vergleichen möchte – dafür ist der Boxplot deutlich besser geeignet (s. Visualisierung 3).

---

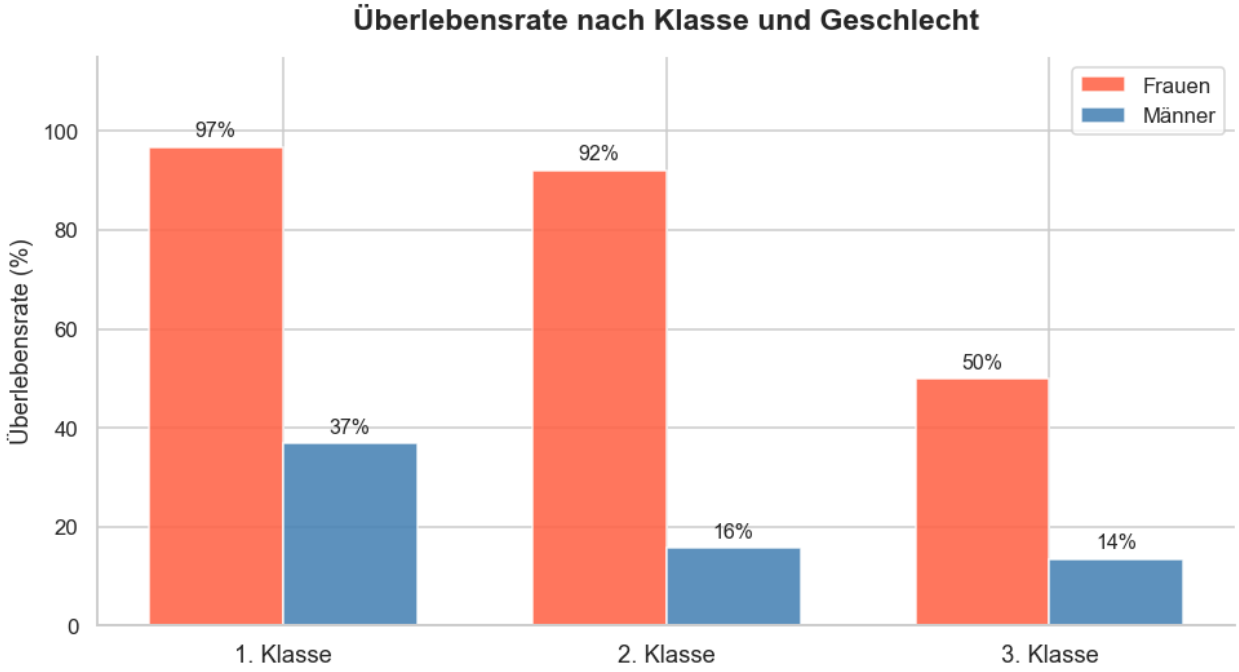
2. Visualisierung: Balkendiagramm

Wann ein Balkendiagramm?

Wenn man **kategorische Gruppen vergleicht** – z.B. Klassen, Geschlecht, Herkunft. Balken machen Unterschiede zwischen Kategorien direkt und präzise ablesbar.

Frage: Wie unterscheidet sich die Überlebensrate nach Reiseklasse und Geschlecht?

In [3]:
```
survival = df.groupby(['pclass', 'sex'])['survived'].mean().reset_index()
survival['survived_pct'] = survival['survived'] * 100

fig, ax = plt.subplots(figsize=(9, 5))

colors_sex = {'male': 'steelblue', 'female': 'tomato'}
labels_sex = {'male': 'Männer', 'female': 'Frauen'}
x = [0, 1, 2]
width = 0.35

for i, (sex, grp) in enumerate(survival.groupby('sex')):
    offset = -width / 2 if i == 0 else width / 2
    bars = ax.bar([xi + offset for xi in x], grp['survived_pct'],
                  width=width, label=labels_sex[sex], color=colors_sex[sex], a
    for bar in bars:
        h = bar.get_height()
        ax.text(bar.get_x() + bar.get_width() / 2, h + 1.2, f'{h:.0f}%',
                ha='center', va='bottom', fontsize=10.5)

ax.set_xticks(x)
ax.set_xticklabels(['1. Klasse', '2. Klasse', '3. Klasse'], fontsize=12)
ax.set_ylabel('Überlebensrate (%)', fontsize=12)
ax.set_ylim(0, 115)
ax.set_title('Überlebensrate nach Klasse und Geschlecht', fontsize=15, fontwei
ax.legend(fontsize=11)

sns.despine()
plt.tight_layout()
plt.show()
```



Beobachtung: Frauen überlebten in allen Klassen deutlich häufiger. Klassenunterschiede sind stark erkennbar. **Warum kein Tortendiagramm?** Tortendiagramme eignen sich nur für Anteile eines Ganzen mit wenigen Kategorien. Für Gruppenvergleiche sind Balken fast immer besser – Längen lassen sich viel präziser ablesen als Winkel.

---

3. Visualisierung: Boxplot

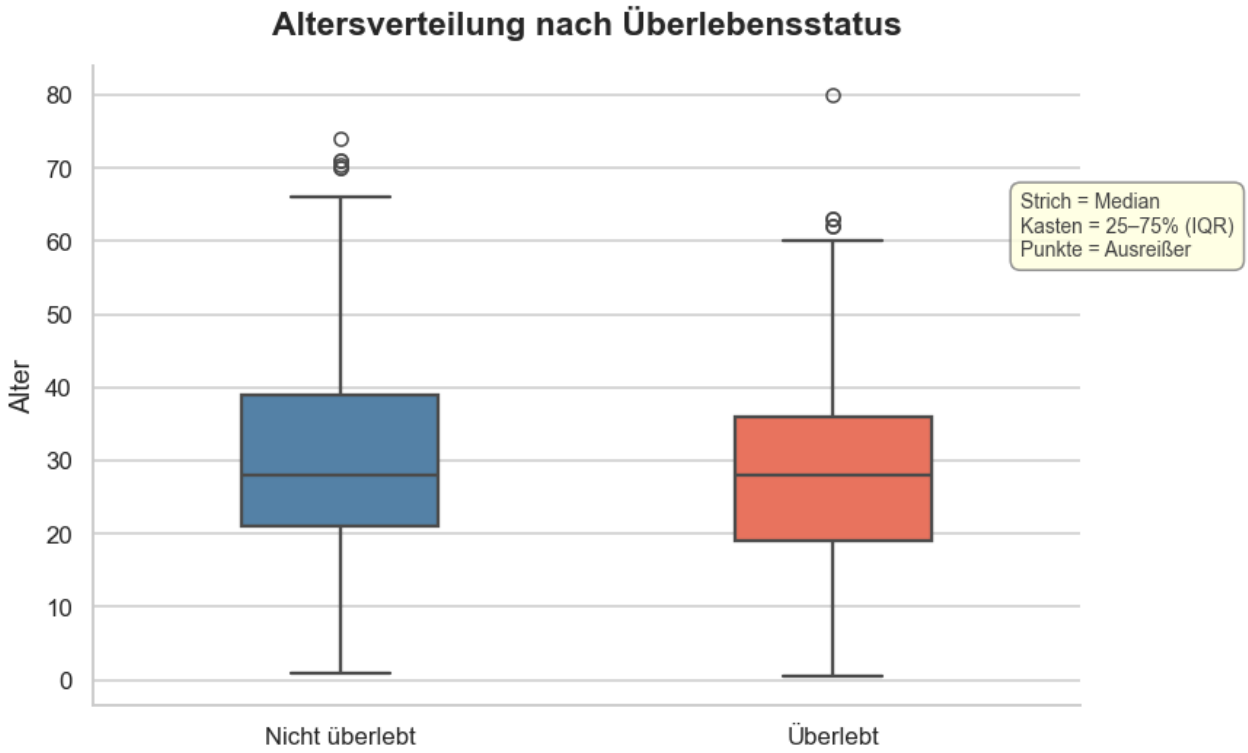
Wann ein Boxplot?

Wenn man eine **numerische Variable über mehrere Gruppen hinweg vergleichen** möchte. Der Boxplot zeigt Median, Streuung (IQR) und Ausreißer auf einmal – ein Histogramm könnte das nur mit mehreren überlagerten Grafiken.

Frage: Unterscheidet sich das Alter zwischen Überlebenden und Nicht-Überlebenden?

In [4]:
```
df_box = df.copy()
df_box['Überleben'] = df_box['survived'].map({0: 'Nicht überlebt', 1: 'Überleb

fig, ax = plt.subplots(figsize=(8, 5))

sns.boxplot(
    data=df_box, x='Überleben', y='age',
    hue='Überleben', legend=False,
    palette={'Nicht überlebt': 'steelblue', 'Überlebt': 'tomato'},
    width=0.4, linewidth=1.5, ax=ax
)

ax.set_title('Altersverteilung nach Überlebensstatus', fontsize=15, fontweight
ax.set_xlabel('', fontsize=12)
ax.set_ylabel('Alter', fontsize=12)

ax.text(1.38, 58,
        'Strich = Median\nKasten = 25–75% (IQR)\nPunkte = Ausreißer',
```

```
        fontsize=9, color='#444',
        bbox=dict(boxstyle='round,pad=0.5', facecolor='lightyellow',
                  edgecolor='gray', alpha=0.85))

sns.despine()
plt.tight_layout()
plt.show()

for gruppe, data in df_box.groupby('Überleben'):
    print(f"{gruppe}: Median = {data['age'].median():.1f} | IQR = {data['age']
```




```
Nicht überlebt: Median = 28.0 | IQR = 21.0 – 39.0
Überlebt: Median = 28.0 | IQR = 19.0 – 36.0
```

Beobachtung: Der Median ist bei beiden Gruppen ähnlich. Kinder tauchen leicht häufiger unter den Überlebenden auf.

Wann kein Boxplot? Bei sehr kleinen Stichproben (< 20 Werte) – dann ist ein Strip- oder Swarmplot ehrlicher, weil er die einzelnen Datenpunkte zeigt.

---

4. Visualisierung: Scatterplot

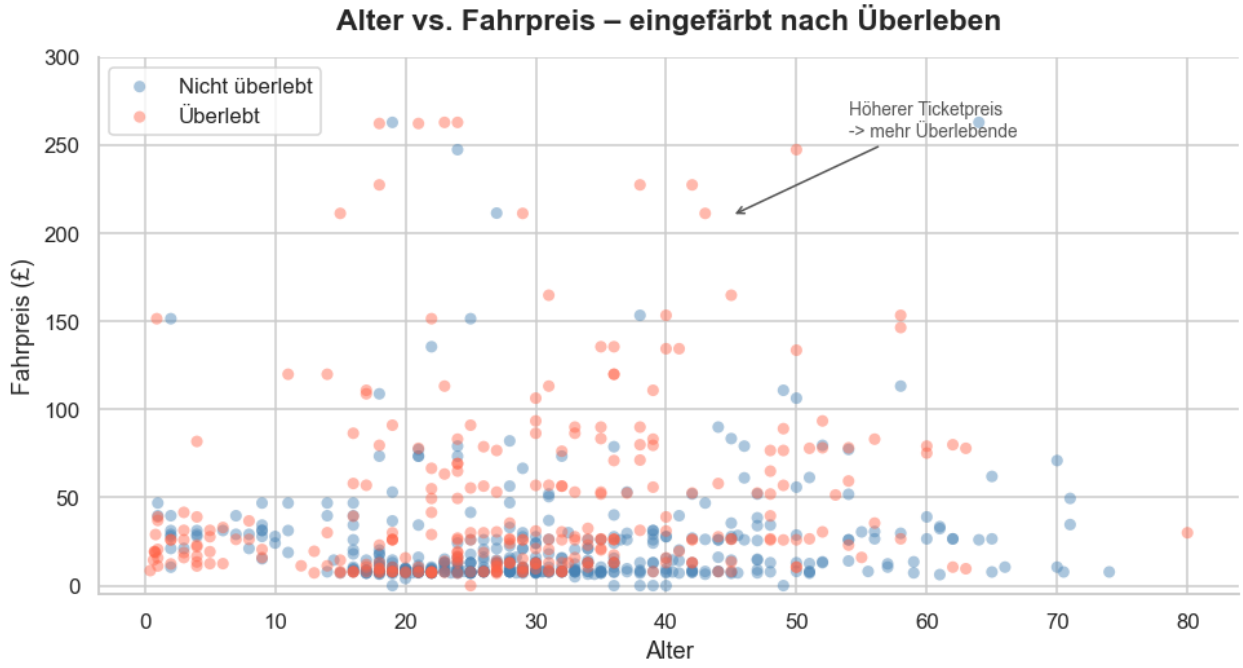
Wann ein Scatterplot?

Wenn man den **Zusammenhang zwischen zwei numerischen Variablen** untersucht. Durch Farbe kann man gleichzeitig eine dritte (kategorische) Variable einbinden – so werden drei Dimensionen in einer Grafik sichtbar.

Frage: Gibt es einen Zusammenhang zwischen Alter, Fahrpreis und Überleben?

In [5]:
```
df_scatter = df.copy()
df_scatter['Überleben'] = df_scatter['survived'].map({0: 'Nicht überlebt', 1:
```

```
fig, ax = plt.subplots(figsize=(9, 5))

colors_map = {'Nicht überlebt': 'steelblue', 'Überlebt': 'tomato'}
for gruppe, grp in df_scatter.groupby('Überleben'):
    ax.scatter(grp['age'], grp['fare'], label=gruppe,
               alpha=0.45, s=35, color=colors_map[gruppe], edgecolors='none')

ax.set_title('Alter vs. Fahrpreis – eingefärbt nach Überleben', fontsize=15, f
ax.set_xlabel('Alter', fontsize=12)
ax.set_ylabel('Fahrpreis (£)', fontsize=12)
ax.set_ylim(-5, 300)
ax.legend(fontsize=11)

ax.annotate('Höherer Ticketpreis \n-> mehr Überlebende',
            xy=(45, 210), xytext=(54, 255),
            fontsize=9, color='#555',
            arrowprops=dict(arrowstyle='->', color='#555'))

sns.despine()
plt.tight_layout()
plt.show()
```



Beobachtung: Passagiere mit teuren Tickets (oben) überlebten häufiger – erkennbar an der höheren Dichte roter Punkte. Bei günstigen Tickets dominieren blaue Punkte.

Wann kein Scatterplot? Bei kategorischen Variablen auf beiden Achsen – dann lieber eine Heatmap oder ein Balkendiagramm.

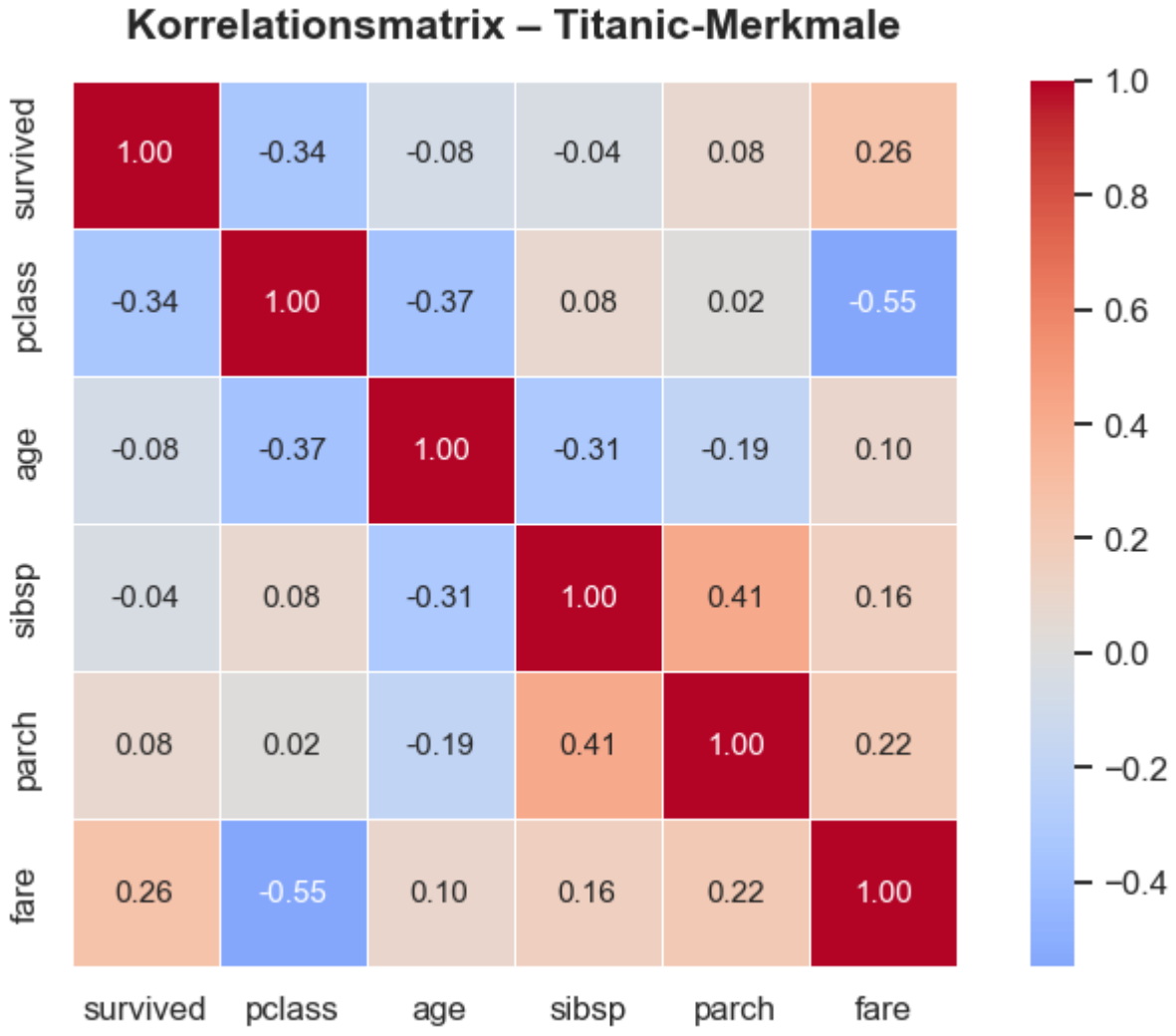
Anhang – Visualisierung 5: Heatmap

Korrelationsmatrix der numerischen Variablen

Wann eine Heatmap?
Wenn man schnell einen **Überblick über Zusammenhänge zwischen vielen Variablen**

bekommen möchte. Besonders nützlich am Anfang einer explorativen Analyse – als Orientierungshilfe, bevor man sich auf einzelne Variablen fokussiert.

In [6]:
```
cols = ['survived', 'pclass', 'age', 'sibsp', 'parch', 'fare']
corr = df[cols].corr()

fig, ax = plt.subplots(figsize=(7, 5))

sns.heatmap(corr, annot=True, fmt='.2f', cmap='coolwarm', center=0,
            linewidths=0.5, square=True, ax=ax, annot_kws={'size': 10})

ax.set_title('Korrelationsmatrix – Titanic-Merkmale', fontsize=14, fontweight=
plt.tight_layout()
plt.show()

print("Stärkste Korrelationen mit 'survived':")
print(corr['survived'].drop('survived').sort_values(key=abs, ascending=False).
```




```
Stärkste Korrelationen mit 'survived':
pclass   -0.338481
fare      0.257307
parch     0.081629
age      -0.077221
sibsp    -0.035322
```

Beobachtung: `pclass` hat die stärkste (negative) Korrelation mit `survived`. `fare` korreliert positiv. Das deckt sich mit dem, was wir im Scatterplot gesehen haben – die

Heatmap gibt den schnellen Überblick.

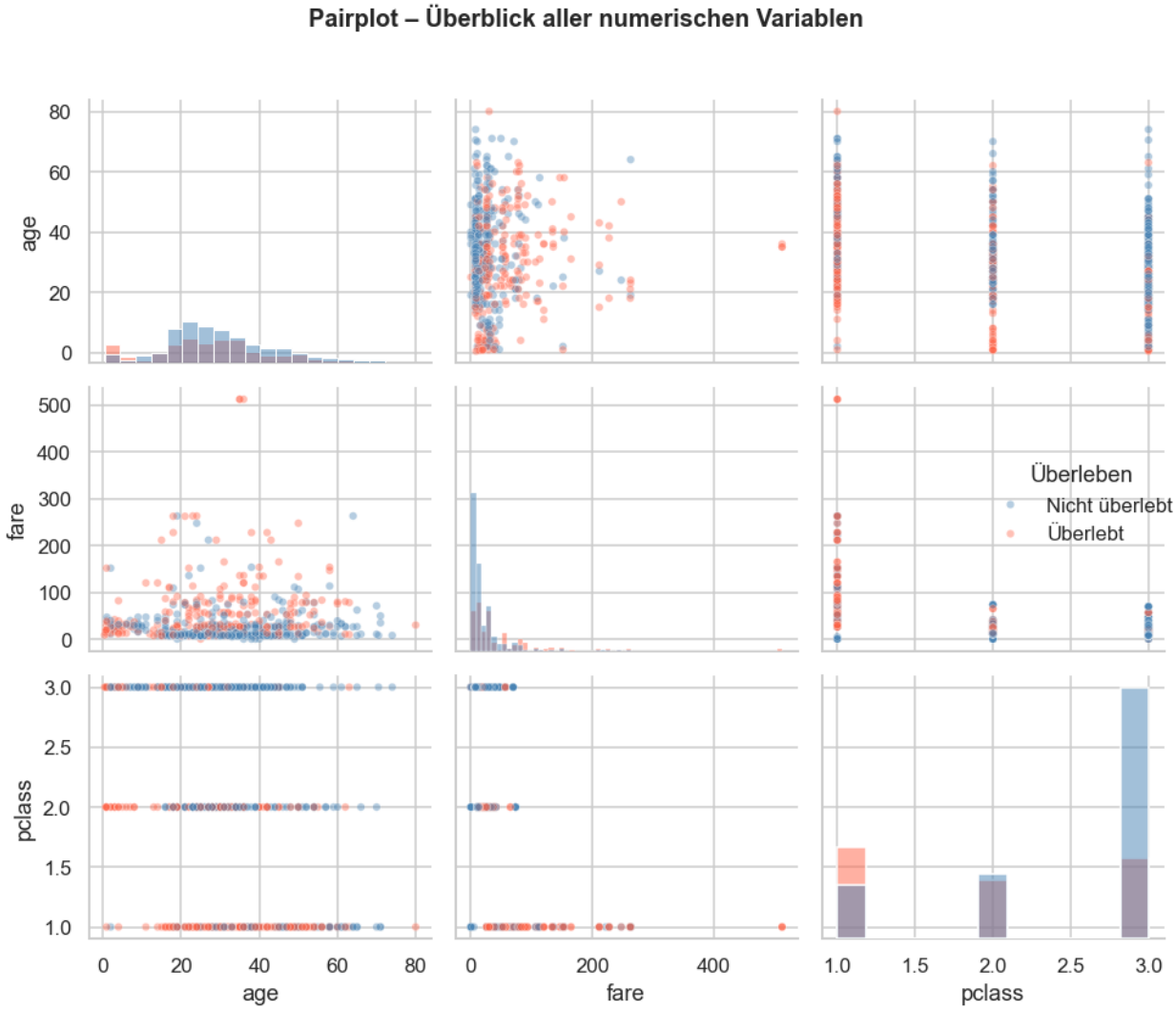
Anhang – Visualisierung 6: Pairplot

Der folgende Python-Code im Anhang für Visualisierung Nr.6 wurde mithilfe von KI generiert und überprüft. Modell: Claude Sonnet 4.6.

Alle numerischen Variablen auf einen Blick

Wann ein Pairplot?
Nur in der **frühen Explorationsphase**, wenn man noch gar nicht weiß, welche Variablen interessant sind. Für Berichte oder Präsentationen ist er meist zu unübersichtlich – er ist ein Werkzeug für den Analysten, nicht für das Publikum.

In [7]:
```
"""
Der folgende Code für das Pairplot wurde mit Hilfe von KI generiert.
Modell: Claude 4.6 Sonnet
"""

df_pair = df[['survived', 'age', 'fare', 'pclass']].copy()
df_pair['Überleben'] = df_pair['survived'].map({0: 'Nicht überlebt', 1: 'Überl

g = sns.pairplot(
    df_pair[['age', 'fare', 'pclass', 'Überleben']],
    hue='Überleben',
    palette={'Nicht überlebt': 'steelblue', 'Überlebt': 'tomato'},
    diag_kind='hist',
    plot_kws={'alpha': 0.4, 's': 18}
)

g.figure.suptitle('Pairplot – Überblick aller numerischen Variablen',
                  y=1.02, fontsize=13, fontweight='bold')
plt.tight_layout()
plt.show()
```



Fazit: Der Pairplot ist ein Tool für die eigene Analyse – nicht für die Kommunikation von Ergebnissen. Sobald man weiß, was interessant ist, greift man auf gezielte Einzelplots zurück.

Zusammenfassung:

Ziel	Visualisierung
Verteilung **einer** numerischen Variable	**Histogramm**
Unterschiede nach **Kategorien**	**Balkendiagramm**
Verteilung zwischen **Gruppen** vergleichen	**Boxplot**
Zusammenhänge zwischen zwei Variablen erkennen	**Scatterplot**
Korrelation zweier Variablen	**Heatmap**
Interessante Variablen entdecken	**Pairplot** (nur Exploration)