Gál Emese

Gruppe: 721

Shelter Animal Outcomes

Prognose-Verbesserung für Tiere in Tierheim

Beschreibung des Datensatzes

Jedes Jahr landen Millionen von Haustieren in US Tierheimen. Viele Tiere wurden von ihren Besitzern aufgegeben, da sie nicht mehr erwünscht waren. Andere Tiere wurden von grausamen Situationen herausgeholt. Die Tierheime hoffen, dass die Haustiere eine liebevolle Familie finden, leider passiert das nicht immer.

Daten

Informationen über die Tieren vom Austin Animal Center

AnimalID	Name	Breed
AnimalType	DateTime	Color

OutcomeType	OutcomeSubtype
SexuponOutcome	AgeuponOutcome

Zielstellung

OutcomeType – Prognose für die Zukunft der Tiere bestimmen, Trends kennenlernen

- macht sichbar welche Tiere mehr Unterstützung brauchen, um adoptiert zu werden

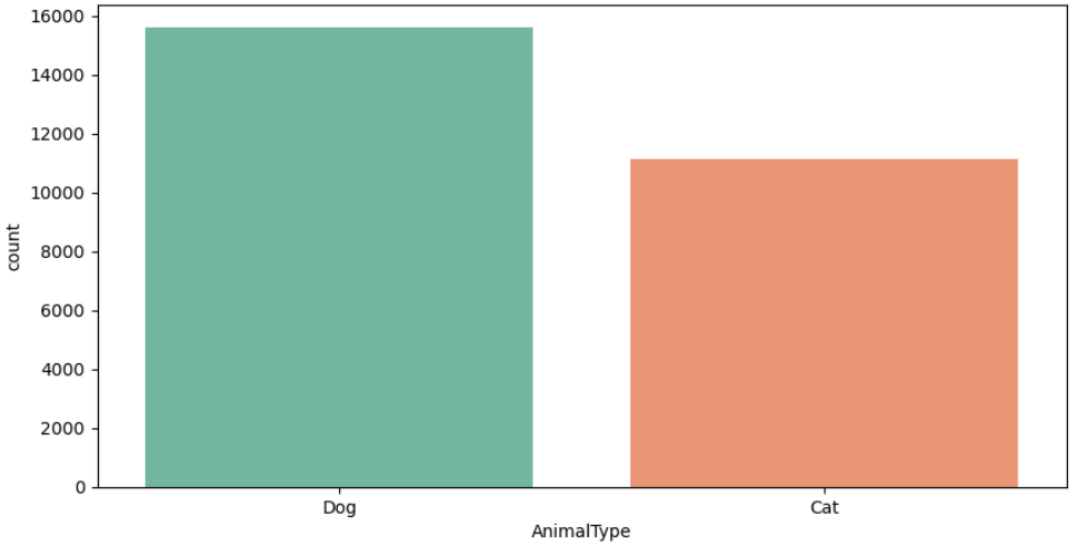
---

Train-Daten mit Plots anschauen

AnimalType – 2 Kategorien:

Hund	Katze

```
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(10, 5))
_ = sns.countplot(data=data, x='AnimalType', palette='Set2')
```

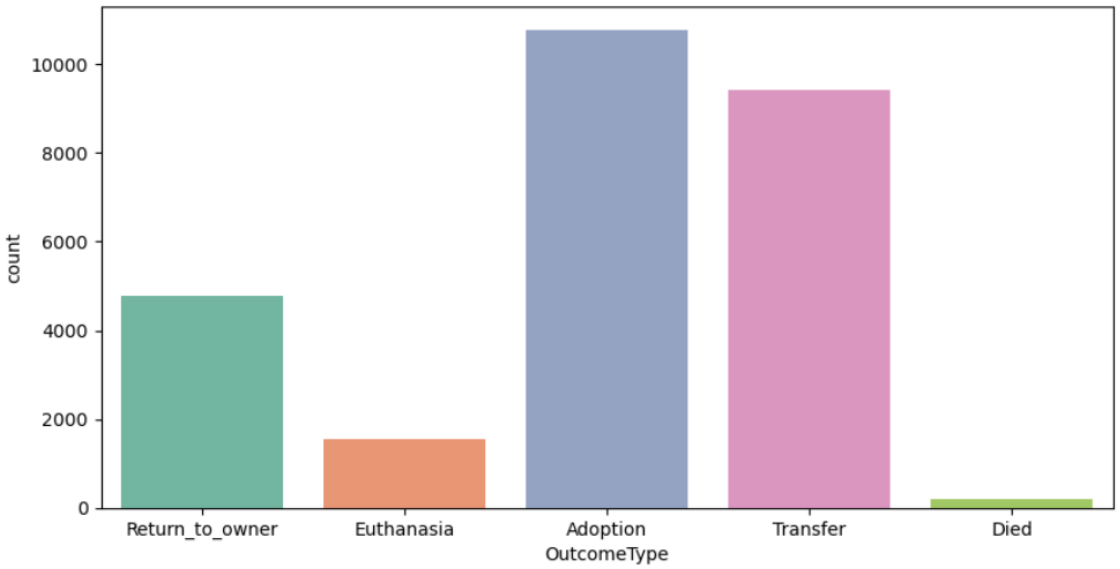


- bei dem Tierheim sind mehrere Hunde

OutcomeType – 5 Kategorien:

zum Besitzer zurückgeben	Euthanasie	Adoption	Transfer	Tot

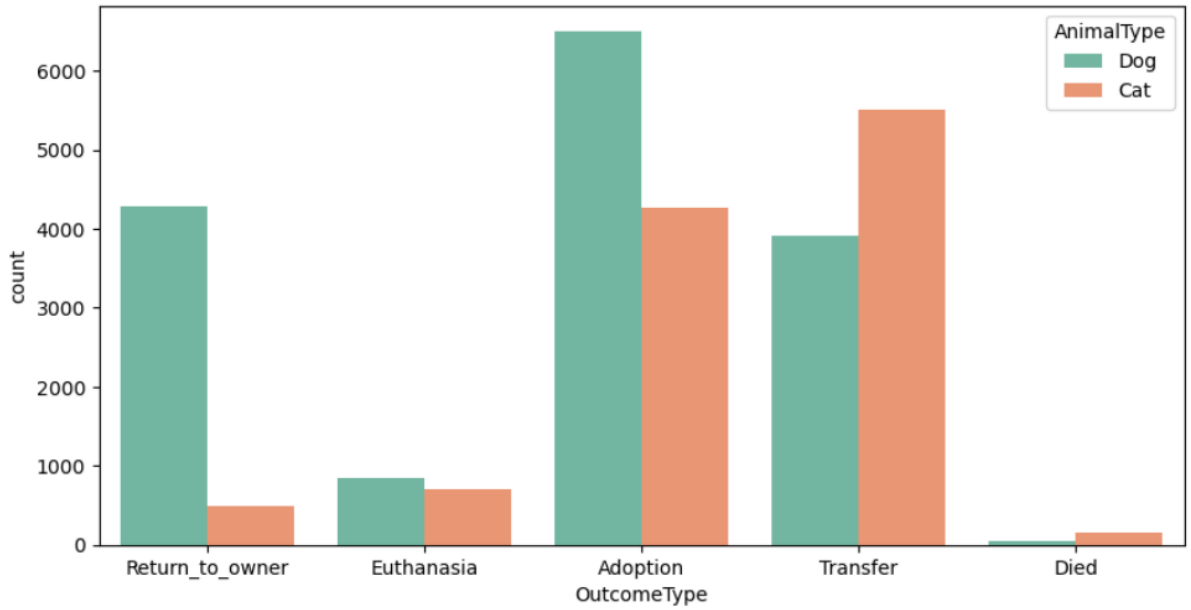
```
plt.figure(figsize=(10, 5))
_ = sns.countplot(data=data, x='OutcomeType', palette='Set2')
```



- Adoptionen bzw. Transfers werden oft gemacht, aber die Anzahl der Fälle von Euthanasie ist nicht unerheblich!

Ändern sich die Trends, wenn man nur Hunde und Katzen betrachtet?

```
plt.figure(figsize=(10, 5))
_ = sns.countplot(data=data, x='OutcomeType', hue="AnimalType", palette='Set2')
```



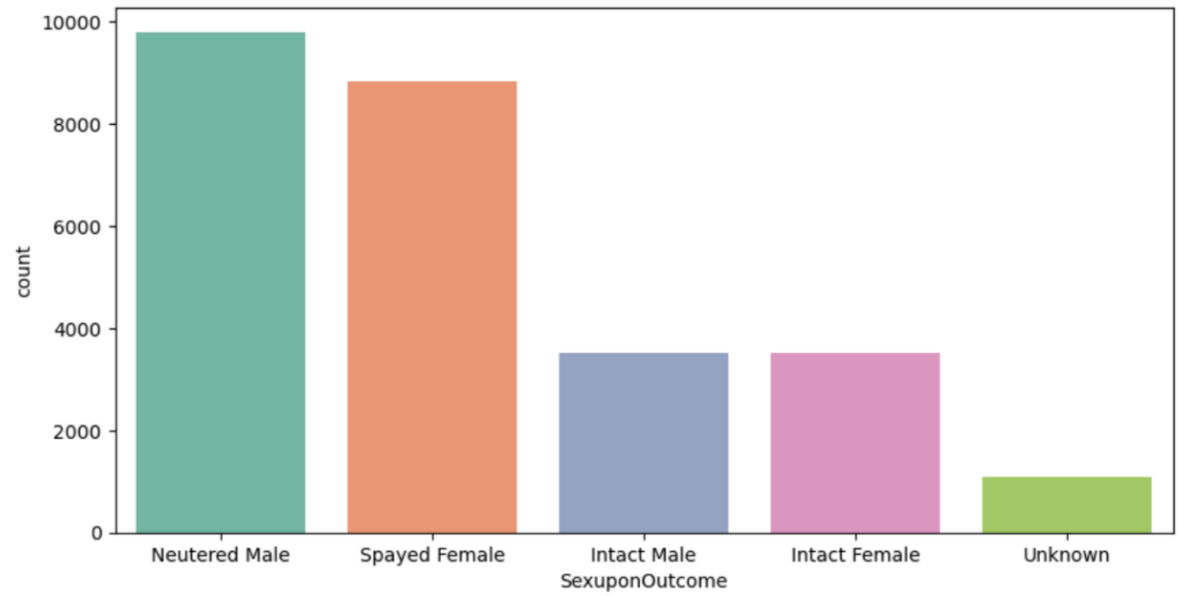
- Hunde wurden oft dem Besetzer zurückgegeben, da sie verloren wurden, während bei den Katzen der Transfer der häufigste ist.

SexuponOutcome – 5 Kategorien

Sterilierste männlich	Sterilisierte weiblich	Intakt männlich	Intakt weiblich	Unbekannt

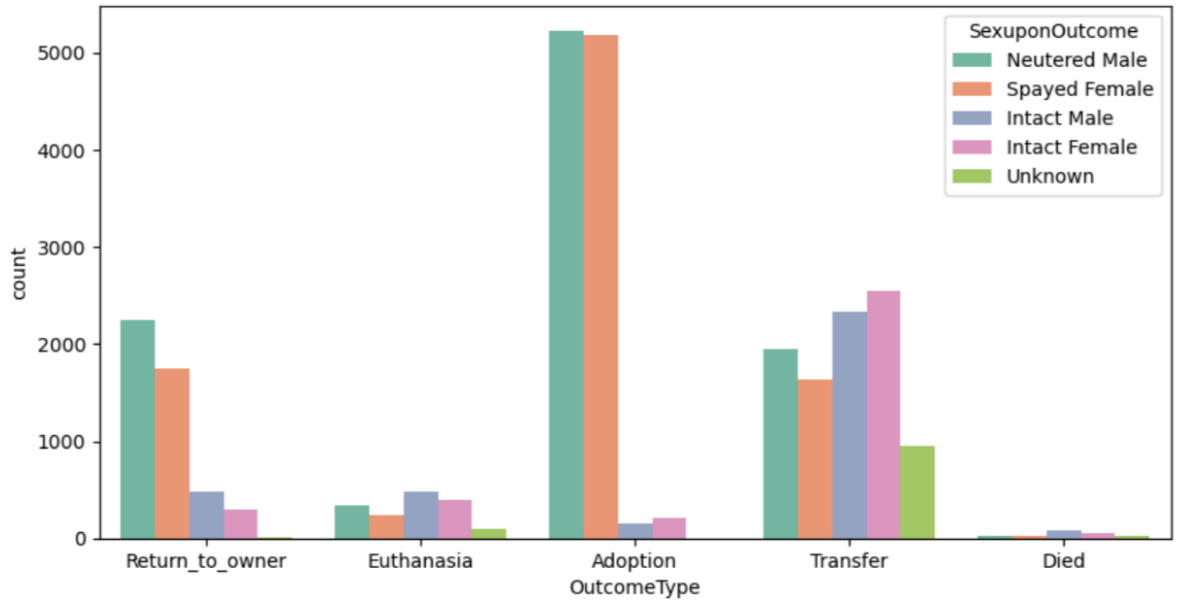
- Die Tiere im Tierheim sind meistens sterilisiert

```
plt.figure(figsize=(10, 5))
_ = sns.countplot(data=data, x='SexuponOutcome', palette='Set2')
```



Was ist der Zusammenhang zwischen Geschlecht und OutcomeTypen?

```
plt.figure(figsize=(10, 5))
_ = sns.countplot(data=data, x='OutcomeType', hue="SexuponOutcome", palette='Set2')
```



- Fast alle adoptierten Tiere sind sterilisiert, die intakten Tiere sind meistens transferiert

Baseline-Modell – Link zum Notebook hier (Version 2)

Die *Shelter Animal Outcomes* Aufgabe ist eine Multiclass Klassifikations-Aufgabe, denn das Ziel ist eine Outcome aus einer festgelegten Menge von Werten vorherzusagen (dem Besitzer zurückgeben, Euthanasie, Adoption, Transfer, Sterben).

Dafür es ist notwendig numerische Daten zu haben. Die Konversion ist einfach mit der Hilfe von LabelEncoder().fit_transform(..). Alle Merkmale sind enthalten, außer die Namen, weil adoptierten Haustiere oft neue Namen bekommen.

AnimalID ist nicht erheblich die Outcome zu bestimmen, deshalb wird es im Training nicht enthalten sein (als Feature). OutcomeSubtype hängt von Outcome ab, deswegen wird es ebenfalls ignoriert.

Als einfache Standardmethode, wird RandomForestclassifier verwendet. Die angegebene Daten waren in 75-25% verteilt, für das eigentliche Training, bwz. für das Testen.

Die Genauigkeit von die 25% anhand des Models ist: 59.91 %, während sie für die 75% der Trainingsdaten 99.99%. (bei einer Ausführung)

```
Accuracy für getestete Werte:  59.91321262905881 %
Accuracy für eigentliche Trainingsdaten:  99.99002294722139 %
```

Potentielles Problem: **Overfitting**

Performanz erhöhen

Cross Validation – für 3 Klassifikatoren

DecisionTreeClassifier	LinearDiscriminantAnalysis	QuadraticDiscriminantAnalysis

durschnittliche Genauigkeiten:

```
DecisionTreeClassifier:  54.11163494302309 %
LinearDiscriminantAnalysis:  52.348688097089024 %
QuadraticDiscriminantAnalysis:  58.41717936476915 %
```

➔ die beste Wahl: QuadraticDiscriminantAnalysis

(vergeblich hatten wir vorher 59.91% für DecisionsTreeClassifier) – durschnittlicher Wert ist mehr von Bedeutung

bei einer Ausführung die Genauigkeit bei dem Model mit QuadraticDiscriminantAnalysis:

```
Accuracy für getestete Werte:  58.416878647314086 %
Accuracy für eigentliche Trainingsdaten:  58.78978349795471 %
```

Ideen für die Verbesserung - Link zum Notebook hier (Version 3)

```python
# eindeutige Elemente
pd.DataFrame([(col, len(data[col].unique())) for col in data.columns])
```

	0	1
0	AnimalID	26729
1	Name	6375
2	DateTime	22918
3	OutcomeType	5
4	OutcomeSubtype	17
5	AnimalType	2
6	SexuponOutcome	6
7	AgeuponOutcome	45
8	Breed	1380
9	Color	366

Bei AnimalType, SexuponOutcome, OutcomeType, Color gibt es wenige eindeutige Elemente.

In den anderen Fällen können wir die Werte vereinfachen:

- Name – besitzt Name oder nicht
- DateTime – in Jahreszeit, Jahr, Wochentag verteilen
- AgeuponOutcome – in Tagen umwandeln
- Breed – Reinrassige oder Mischlinge

Dafür benutzen wir List-Comprehensions.

Numerische Bedeutungen

Outcome

0	1	2	3	4
Adoption	Died	Euthanasia	Return to owner	Transfer

	Name	AnimalType	SexuponOutcome	AgeuponOutcome	Breed	Color	Year	Season	WeekDay
0	1	1	2	356	0	130	2014	3	6
1	1	0	3	356	0	167	2013	2	3
2	1	1	2	712	0	86	2015	3	2
3	0	0	1	21	0	42	2014	1	0
4	0	1	2	712	1	274	2013	2	0

Name

0	1
besitzt Name	besitzt keinen Name

AnimalType

0	1
Cat	Dog

SexuponOutcome

0	1	2	3	4
Intact Female	Intact Male	Neutered Male	Spayed Female	Unknown

Breed

0	1
Mischling	Reinressig

Season – abhängig von Monatnummer

3-5	6-8	9-11	12,1,2
Frühling	Sommer	Herbst	Winter

AgeuponOutcome – Alter in Tagenummer

Color – Jede Farbe hat eine Nummer

Year – Jahr

WeekDay - erster Tag der Woche = 0, ..., letzter Tag der Woche = 6

Problematische nan-Werte

Weil wenige nan-Werte in den Merkmalen sind, werden diese Daten herausgenommen. Wir behalten nur die Datensätze die keine nan-Werte haben in Features.

(mit Hilfe von .notna())

```
data.isnull().sum()
```

```
AnimalID              0
Name               7691
DateTime              0
OutcomeSubtype    13612
AgeuponOutcome       18
Breed                 0
Color                 0
OutcomeType           0
AnimalType            0
SexuponOutcome        0
dtype: int64
```

Hyperparameter Optimization

- für DecisionTreeClassifier die beste Parameter-Kombination:

```
DecisionTreeClassifier(criterion='entropy', max_depth=10, max_features='log2',
                       min_samples_split=100)
```

Cross Validation – für 3 Klassifikatioren

(DecisionTreeClassifier mit die gefundene beste Parameter)

durschnittliche Genauigkeiten:

```
DecisionTreeClassifier:  63.94722604235399 %
LinearDiscriminantAnalysis:  55.34634124687333 %
QuadraticDiscriminantAnalysis:  58.123388802162346 %
```

➔ die beste Wahl: DecisionTreeClassifier

Konklusion

Mit das gemachte Modell ungefähr 64 % Genaugkeit ist erreichbar.