

Лабораторная работа №4. Байесовские сети

ГОРБАН АРТЕМИЙ М8О-307Б-23

DATASET: ZOO ANIMAL CLASSIFICATION

Цель работы

Построить байесовскую сеть с помощью pgmpy.

Оценить условные вероятности (CPT).

Сравнить с Naïve Bayes.

Данные

101 животное, 18 признаков, 7 классов (млекопитающие, птицы и др.).

Признаки: hair, feathers, milk, aquatic, legs и т.д. (все бинарные или дискретные).

Обработка: Закодировали названия, удалили дубликаты.

```
binary_cols = ['hair', 'feathers', 'eggs', 'milk', 'airborne', 'aquatic', 'predator',
               'toothed', 'backbone', 'breathes', 'venomous', 'fins', 'tail',
               'domestic', 'catalize']

for col in binary_cols:
    unique_vals = sorted(data[col].unique())
    if set(unique_vals) != {0, 1}:
        print(f"{col} имеет значения {unique_vals}")
    else:
        print(f"{col} = {unique_vals}")

print(f"legs: {sorted(data['legs'].unique())}")
print(f"class_type: {sorted(data['class_type'].unique())}")

hair = [np.int64(0), np.int64(1)]
feathers = [np.int64(0), np.int64(1)]
eggs = [np.int64(0), np.int64(1)]
milk = [np.int64(0), np.int64(1)]
airborne = [np.int64(0), np.int64(1)]
aquatic = [np.int64(0), np.int64(1)]
predator = [np.int64(0), np.int64(1)]
toothed = [np.int64(0), np.int64(1)]
backbone = [np.int64(0), np.int64(1)]
breathes = [np.int64(0), np.int64(1)]
venomous = [np.int64(0), np.int64(1)]
fins = [np.int64(0), np.int64(1)]
tail = [np.int64(0), np.int64(1)]
domestic = [np.int64(0), np.int64(1)]
catalize = [np.int64(0), np.int64(1)]

legs = [np.int64(0), np.int64(2), np.int64(4), np.int64(5), np.int64(6), np.int64(8)]
class_type = [np.int64(1), np.int64(2), np.int64(3), np.int64(4), np.int64(5), np.int64(6), np.int64(7)]
```

Структура сети (HillClimbSearch)

Автоматический поиск зависимостей.
Сеть выявляет биологически логичные связи.

```
from pgmpy.models import DiscreteBayesianNetwork
```

```
# Задаем направление между вершинами
```

```
network = [  
    ('hair', 'class_type'),  
    ('feathers', 'class_type'),  
    ('eggs', 'class_type'),  
    ('milk', 'class_type'),  
    ('airborne', 'class_type'),  
    ('aquatic', 'class_type'),  
    ('predator', 'class_type'),  
    ('toothed', 'class_type'),  
    ('backbone', 'class_type'),  
    ('breathes', 'class_type'),  
    ('venomous', 'class_type'),  
    ('fins', 'class_type'),  
    ('legs', 'class_type'),  
    ('tail', 'class_type'),  
    ('domestic', 'class_type'),  
    ('catsize', 'class_type')  
]
```

```
# Строим Дискретную Байесовскую сеть
```

```
model = DiscreteBayesianNetwork(network)
```

```
model.edges() # Просмотр ребер
```

Python

```
OutEdgeView([('hair', 'class_type'), ('feathers', 'class_type'), ('eggs', 'class_type'), ('milk', 'class_type'), ('airborne', 'class_type'), ('aquatic', 'class_type'), ('predator', 'class_type'), ('toothed', 'class_type'), ('backbone', 'class_type'), ('breathes', 'class_type'), ('venomous', 'class_type'), ('fins', 'class_type'), ('legs', 'class_type'), ('tail', 'class_type'), ('domestic', 'class_type'), ('catsize', 'class_type')])
```

Оценка вероятностей (CPT)

Использовали BayesianEstimator.

CPT показывает вероятности признаков при условии родительских узлов.

```
cpt_class = model.get_cpds('class_type')
print(cpt_class)
```

```
+-----+-----+-----+
| toothed | toothed(0) | toothed(1) |
+-----+-----+-----+
| class_type(1) | 0.03809523809523809 | 0.616883116883117 |
+-----+-----+-----+
| class_type(2) | 0.4603174603174603 | 0.010822510822510826 |
+-----+-----+-----+
| class_type(3) | 0.03809523809523809 | 0.07142857142857145 |
+-----+-----+-----+
| class_type(4) | 0.015873015873015872 | 0.20779220779220783 |
+-----+-----+-----+
| class_type(5) | 0.015873015873015872 | 0.07142857142857145 |
+-----+-----+-----+
| class_type(6) | 0.1936507936507936 | 0.010822510822510826 |
+-----+-----+-----+
| class_type(7) | 0.23809523809523805 | 0.010822510822510826 |
+-----+-----+-----+
```

```
for node in ['hair', 'feathers', 'milk']:
    cpt = model.get_cpds(node)
    print(f"CPT for {node}:\n{cpt}")
```

CPT for hair:

```
+-----+-----+-----+
| aquatic | aquatic(0) | ... | aquatic(1) |
+-----+-----+-----+
| milk | milk(0) | ... | milk(1) |
+-----+-----+-----+
| hair(0) | 0.8384615384615385 | ... | 0.38235294117647056 |
+-----+-----+-----+
| hair(1) | 0.16153846153846155 | ... | 0.6176470588235294 |
+-----+-----+-----+
```

CPT for feathers:

```
+-----+-----+-----+
| class_type | ... | class_type(7) |
+-----+-----+-----+
| feathers(0) | ... | 0.9375000000000001 |
+-----+-----+-----+
| feathers(1) | ... | 0.06250000000000001 |
+-----+-----+-----+
```

CPT for milk:

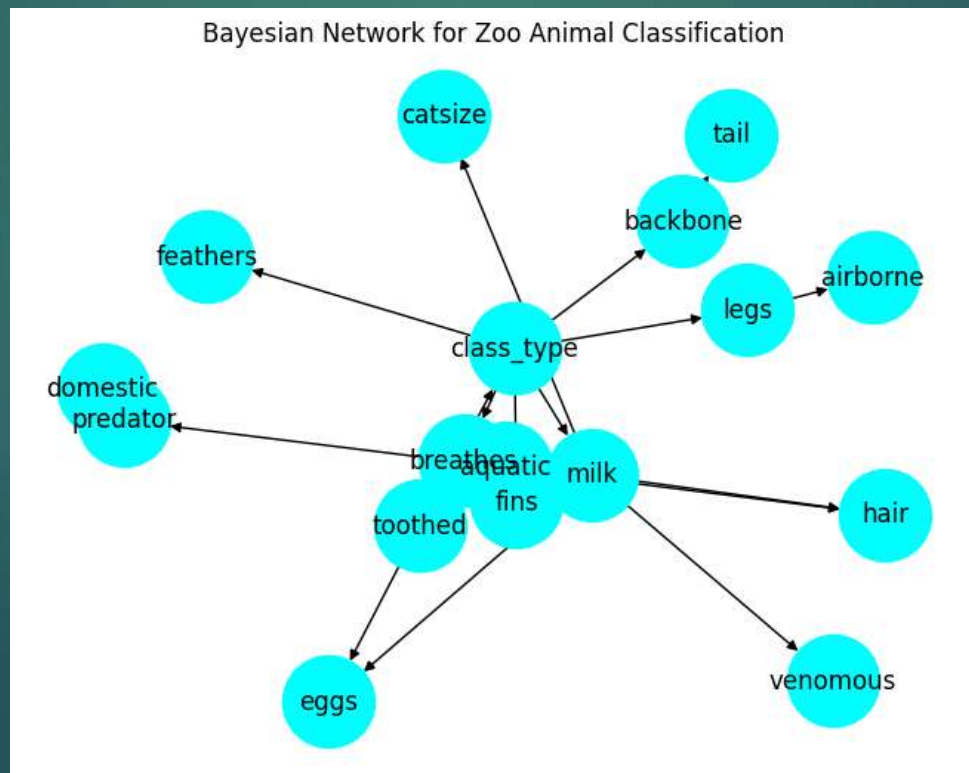
```
+-----+-----+-----+
| class_type | class_type(1) | ... | class_type(7) |
+-----+-----+-----+
| milk(0) | 0.016835016835016835 | ... | 0.9375000000000001 |
+-----+-----+-----+
| milk(1) | 0.9831649831649831 | ... | 0.06250000000000001 |
+-----+-----+-----+
```

Визуализация сети

Нарисовали граф (узлы - признаки, стрелки - зависимости).

Центральный узел - class_type.

Видно, как признаки влияют друг на друга (например, milk влияет на hair и eggs).



ЛОГИЧЕСКИЙ ВЫВОД (Inference)

Вычисление вероятностей при известных признаках.

```
from pgmpy.inference import VariableElimination

infer = VariableElimination(model)
query = infer.query(variables=['class_type'], evidence={'hair': 1, 'milk': 1})
print(query)
```

```
+-----+-----+
| class_type | phi(class_type) |
+=====+=====+
| class_type(1) | 0.9136 |
+-----+-----+
| class_type(2) | 0.0159 |
+-----+-----+
| class_type(3) | 0.0148 |
+-----+-----+
| class_type(4) | 0.0116 |
+-----+-----+
| class_type(5) | 0.0152 |
+-----+-----+
| class_type(6) | 0.0156 |
+-----+-----+
| class_type(7) | 0.0134 |
+-----+-----+
```

Сравнение с Naïve Bayes

Обе модели показали точность 100% на тестовых примерах.
Разница: Байесовская сеть показывает связи между признаками, а Naïve Bayes предполагает их независимость.
Итог: Точность одинаковая, но байесовская сеть лучше для понимания данных.

ВЫВОДЫ

Байесовская сеть построена и работает.

Её структура имеет биологический смысл.

Модель объясняет, как признаки влияют на класс животного.

Главный плюс — наглядность и интерпретируемость.