Applications of Bayesian Belief Networks

Michał Warzecha Oscar Teeninga

16 stycznia 2022

1 Wstęp

Sieć bayesowska to model graficzny, który koduje probabilistyczne relacje między zmiennymi będącymi przedmiotem zainteresowania. W połączeniu z technikami statystycznymi model graficzny ma kilka zalet w zakresie analizy danych. Po pierwsze, ponieważ model koduje zależności między wszystkimi zmiennymi, z łatwością radzi sobie z sytuacjami, w których: brakuje niektórych wpisów danych. Po drugie, sieć bayesowska może być wykorzystana do uczenia się związków przyczynowych, a zatem może być wykorzystana do zrozumienia domeny problemu i przewidywania konsekwencji interwencji. Po trzecie, ponieważ model ma zarówno semantykę przyczynową, jak i probabilistyczną, jest idealną reprezentacją do łączenia wcześniejszej wiedzy (która często występuje w formie przyczynowej) i danych. W tym artykule omówimy zasady działania algorytmu sieci bayesowskich oraz ich zastosowania w świecie rzeczywistym, które stanowi bardzo małą, ale jednocześnie wyrafinowaną część w dziedzinie machine learningu (1).

Rysunek 1: Rodzaje machine learningu [1])

2 Twierdzenie Bayesa

Działanie sieci bayesowskich opiera się na twierdzeniu Bayesa. Jest to twierdzenie wiążące prawdopodobieństwa warunkowe dwóch zdarzeń warunkujących się nawzajem [4]. W podstawowej wersji twierdzenie wygląda następująco(1):

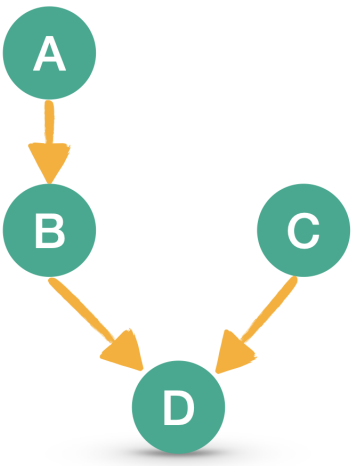
$$P(A \mid B) = \frac{P(B \mid A)\,P(A)}{P(B)} \tag{1}$$

Gdzie A i B są zdarzeniami oraz $P(B) > 0$, przy czym:

- $P(A \mid B)$ oznacza prawdopodobieństwo warunkowe, tj. prawdopodobieństwo zajścia zdarzenia A, o ile zajdzie zdarzenie B.
- $P(B \mid A)$ oznacza prawdopodobieństwo zajścia zdarzenia B, o ile zajdzie zdarzenie A.

3 Sieci Bayesowskie

Sieć bayesowska to probabilistyczny model graficzny, który reprezentuje zbiór zmiennych i ich warunkowe zależności za pomocą skierowanego grafu acyklicznego (DAG) (**??**). BBN buduje acykliczny graf, w którym węzły reprezentują zmienne cech, a połączenia między węzłami reprezentują bezpośrednie probabilistyczną zależność między zmiennymi.



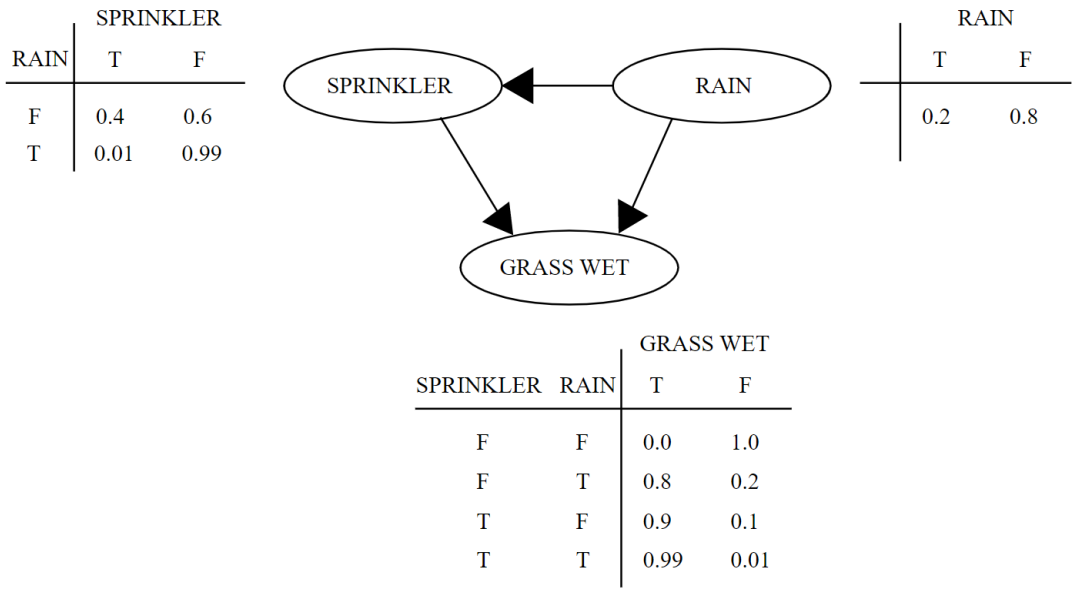
Rysunek 2: Przykładowy skierowany graf acykliczny (drzewo): [1])

Ponieważ sieć jest acyklicznym grafem w którym węzły reprezentują zmienne cechy, natomiast połączenia reprezentują bezpośrednią probabilistyczną zależność między tymi zmiennymi. Jeden węzeł reprezentuje jedną zmienną cech. Każda zmienna musi mieć co najmniej dwa stany dyskretne. W przypadku zmiennej dyskretnej jej wartości cyfrowe lub symboliczne mogą być przekazywane bezpośrednio do stanów węzła. W przypadku zmiennej ciągłej należy ją podzielić na zmienną dyskretną. Każda dyskretna liczba odpowiada jednemu stanowi. Następnie każdy stan jest powiązany z wcześniejszą wartością prawdopodobieństwa; dla każdego węzła suma wartości prawdopodobieństw dla stanów całkowitych wynosi 1. Połączenia między węzłami są reprezentowane przez prawdopodobieństwa warunkowe. Liczbę prawdopodobieństw warunkowych określa struktura sieci, natomiast są one obliczane na podstawie zestawu danych uczących.

Ze względu na budowę grafu (2) możliwe jest stworzenie wielu kombinacji tego jak wyglądać będzie nasz graf. Przykładowo nie każda topologia bayesowska sieci nadaje się do klasyfikacji. Do tego celu można wykorzystać tylko takie grafy, w których klasa jest rodzicem (wszystkich) atrybutów i sama nie ma rodzica. Na przykład w problemach z diagnozą medyczną choroba jest źródłem wszystkich objawów; zatem diagnoza jest rodzicem wszystkich atrybutów (symptomów).

Sieci bayesowskie są idealne do analizy zdarzenia, które miało miejsce i przewidywania prawdopodobieństwa, którego przyczyną była jedna z kilku możliwych znanych przyczyn. Na przykład sieć bayesowska może reprezentować probabilistyczne relacje między chorobami, a objawami. Biorąc pod uwagę objawy, sieć można wykorzystać do obliczenia prawdopodobieństw występowania różnych chorób. [5]

Poniżej widać przykład prostej sieci bayesowskiej(3).



RAIN	SPRINKLER T	SPRINKLER F
F	0.4	0.6
T	0.01	0.99

RAIN T	RAIN F
0.2	0.8

SPRINKLER	RAIN	GRASS WET T	GRASS WET F
F	F	0.0	1.0
F	T	0.8	0.2
T	F	0.9	0.1
T	T	0.99	0.01

Rysunek 3: Przykładowa prosta sieć bayesowska (source: [5])

4 Implementacja Bayesian Belief Network

By przetestować działanie sieci *Bayesian Belief Network* stworzyliśmy przykładową implementację takiej sieci w języku python. W tym celu użyliśmy istniejącej biblioteki *pybbn* odpowiadającego za budowę i trening sieci Bayesa oraz biblioteki *networkx* w celu wizualizacji stworzonego grafu.

4.1 Budowanie sieci

Najważniejszą funkcją, która będzie budować prawdopodobieństwo w węzłach jest funkcja **probs**. Oblicza ona rozkład prawdopodobieństwo dla podanych węzłów.

```
# Funkcja ta pomaga obliczyc
# rozklad prawdopodobienstwa potrzeby do stworzenia BNN

def probs(data, child, parent1=None, parent2=None):
  if parent1==None:
    prob=pd.crosstab(data[child], 'Empty', margins=False,
   normalize='columns').sort_index().to_numpy().reshape(-1).tolist()
  elif parent1!=None:
    if parent2==None:
      prob=pd.crosstab(data[parent1],data[child], margins=False,
   normalize='index').sort_index().to_numpy().reshape(-1).tolist()
    else:
      prob=pd.crosstab([data[parent1],data[parent2]],data[child],
   margins=False,
   normalize='index').sort_index().to_numpy().reshape(-1).tolist()
  else:
    print("Error in Probability Frequency Calculations")

  return prob
```

Algorytm 1: Rozkład prawdopodobieństwa

Aby funkcja działała, należy dostarczyć algorytm generujący rozkład prawdopodobieństwa. Skorzystaliśmy z biblioteki **pandas** oraz funkcji *crosstab* [3]. **crosstab**.

Kolejna funkcja odpowiada za tworzenie węzłów. W tym przypadku korzystamy z faktu, że znamy nazwy danych kategorycznych oraz jesteśmy w stanie na podstawie indeksów kolumn obliczyć za pomocą wyżej wspomnianej funkcji wyznaczającej rozkład prawdopodobieństwo zmienną reprezentowaną przez nowy węzeł.

```
# Generyczny sposob na tworzenie wezlow dla kolumny po indeksie

def create_node(idx, p1=None, p2=None):
  name = df.columns[idx]
  parent1 = df.columns[p1] if p1 != None else None
  parent2 = df.columns[p2] if p2 != None else None
  prob =  probs(df_train, child=name, parent1=parent1, parent2=parent2)
  return BbnNode(Variable(idx, name, [1, 2]), prob)
```

Algorytm 2: Tworzenie węzła

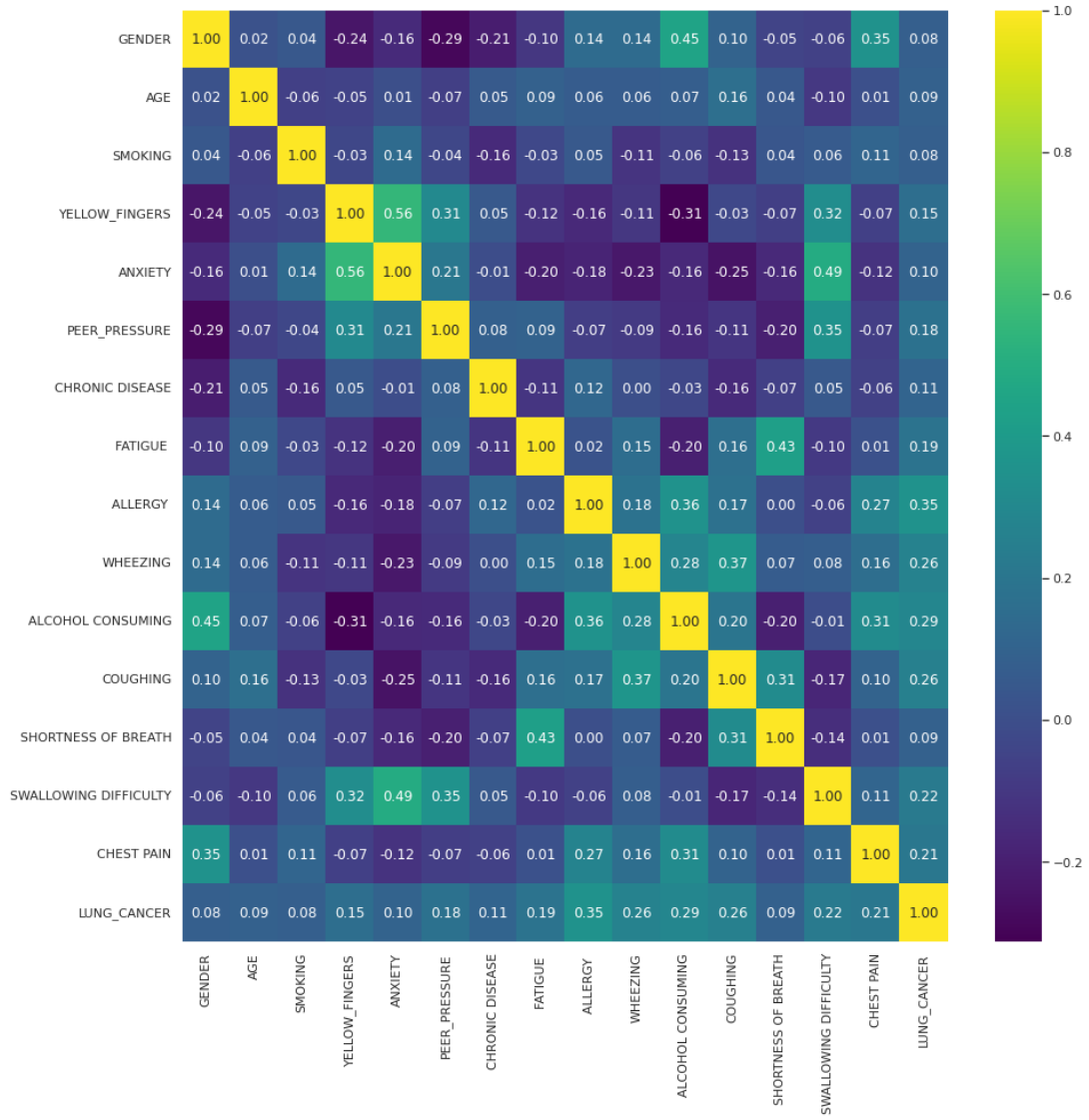
Ostatnim elementem tworzącym sieć jest funkcja która tworzy całą sieć na bazie węzłów i krawędzi. Jest ona prosta i w głównej mierze bazuje na implementacjj dostarczonej przez bibliotekę **pybbn** [5].

```
# Tworzenie sieci, laczenie wezly i krawedzie

def create_network(nodes, edges):
    bbn = Bbn()
    for node in nodes:
        bbn.add_node(node)
    for edge in edges:
        bbn.add_edge(edge)
    return bbn

bbn = create_network(nodes, edges)
```

Algorytm 3: Tworzenie sieci

4.2 Dobór struktury sieci

Aby predykcja przeprowadzana przez sieć była jak najbardziej dokładna, należy odpowiednio zbudować całe drzewo. Aby to zrobić wykorzystujemy wykres korelacji 4, w zależności od danych będzie on różny.

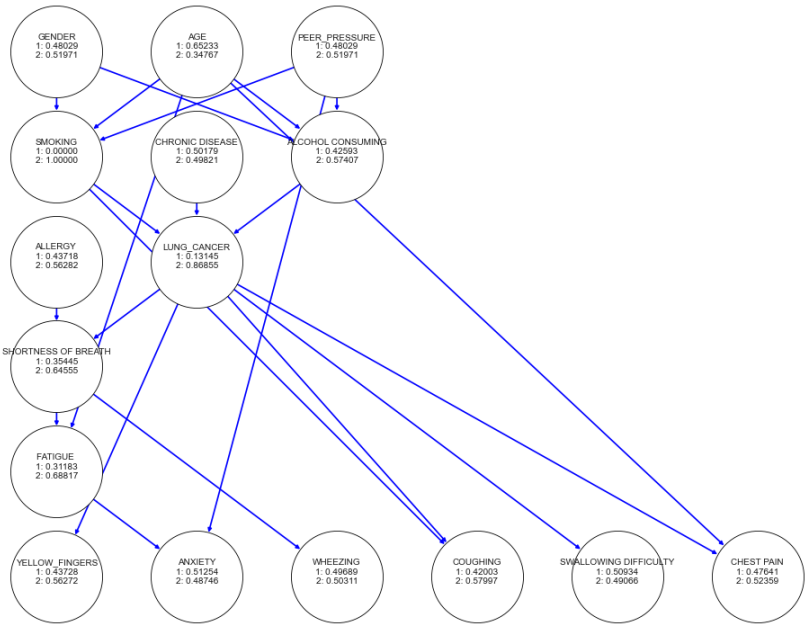


Rysunek 4: Przykładowy wykres korelacji

Przykładowa sieć bayesowska z ręcznie dobranymi krawędziami może wyglądać jak na rysunku 5.

4.3 Automatyczne tworzenie struktury sieci

Inną metodą stworzenia struktury sieci jest przeszukiwanie przestrzeni grafów w poszukiwaniu optymalnej sieci dającej najlepsze rezultaty, Jedną z możliwych metod dla takiego przeszukiwania jest *two-pass greedy search* [2].



Rysunek 5: Przykładowa sieć bayesowska z ręcznie dobranymi krawędziami dla problemu predykcji raka płuc

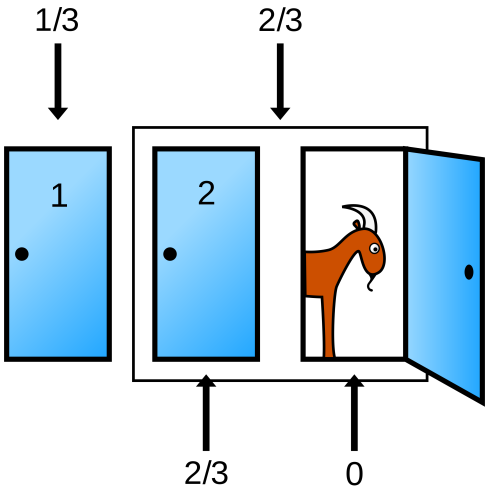
5 Wyniki badań eksperymentalnych

Przeprowadziliśmy kilka eksperymentów, począwszy od prostej sieci rozstrzygającej problem Monty Hall, następnie prosta prognoza pogody dla danych nie-kategorycznych, kończąc na wielo-węzłowej oraz wielo-poziomowej sieci stawiająca diagnozę raka płuc na podstawie danych kategorycznych.

5.1 Problem Monty Hall

Jest to paradoks, który występował w programie *Idź na całość* emitowany w telewizji i prowadzona przez Montiego Halla (od którego powstała nazwa problemu). Gra opierała się na prawdopodobieństwie, natomiast było ono na tyle nieintuicyjne, że nazwano to paradoksem. Zawodnik stoi przed trzema zasłoniętymi bramkami. Za jedną z nich (za którą – wie to tylko prowadzący program) jest nagroda (umieszczana całkowicie losowo). Gracz wybiera jedną z bramek. Prowadzący program odsłania inną bramkę (co istotne – anonsując, że jest to bramka pusta), po czym proponuje graczowi zmianę wyboru. Intuicyjnie nie ma znaczenia, czy zawodnik pozostanie przy swoim wyborze, czy nie. Okazuje się jednak, że jest inaczej. Przy wyborze strategii pozostawania przy swoim pierwszym wyborze prawdopodobieństwo wygranej wynosi 1/3. Natomiast przy wyborze „strategii zmiany" wynosi 2/3.

Oznacza to, że zawodnikowi opłaci się zmienić bramkę, ponieważ ma wtedy dwa razy większe szanse na wygraną. Paradoks wynika z niedocenienia informacji, jaką „między wierszami" przekazuje prowadzący. Informacją tą jest wskazanie pustej bramki. Korzystając z sieci, chcie-

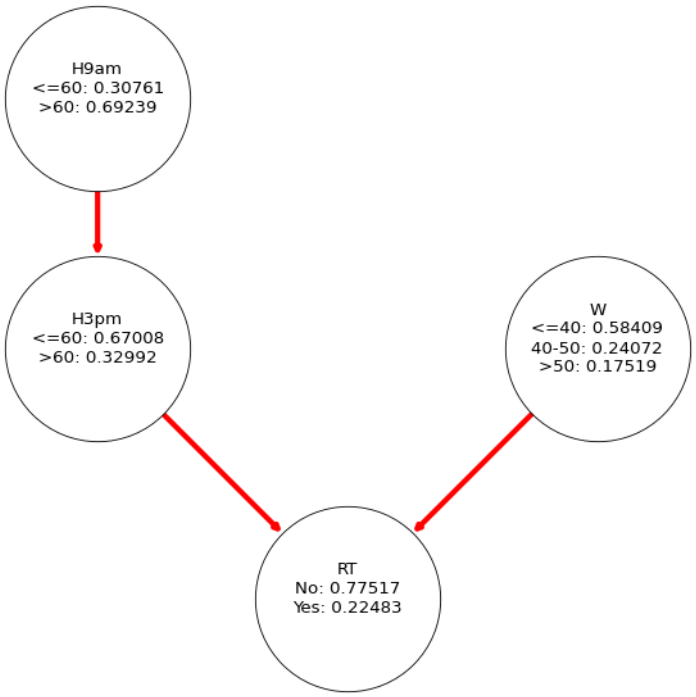


Rysunek 6: Problem Monty Hall

liśmy sprawdzić, czy rzeczywiście bardziej opłacalną strategią jest zmiana wybranej bramki, co udało nam się.

5.2 Prognoza pogody

Skorzystaliśmy ze zbioru dostępnego na platformie kaggle. Są to dane, które dotyczą pogody w Australii, których targetem kategorycznym jest odpowiedź na pytanie, czy jutro będzie padać. Zbiór składa się z wielu cech, natomiast my na potrzebę zbudowania prostej sieci skorzystaliśmy z prędkości wiatru, wilgotności o 9:00, wilgotności o 15:00, ciśnieniu o 9:00 oraz ciśnieniu i 15:00. Zamieniliśmy wszystkie kolumny na dane kategoryczne oraz na ich postawie zbudowaliśmy sieć (7).

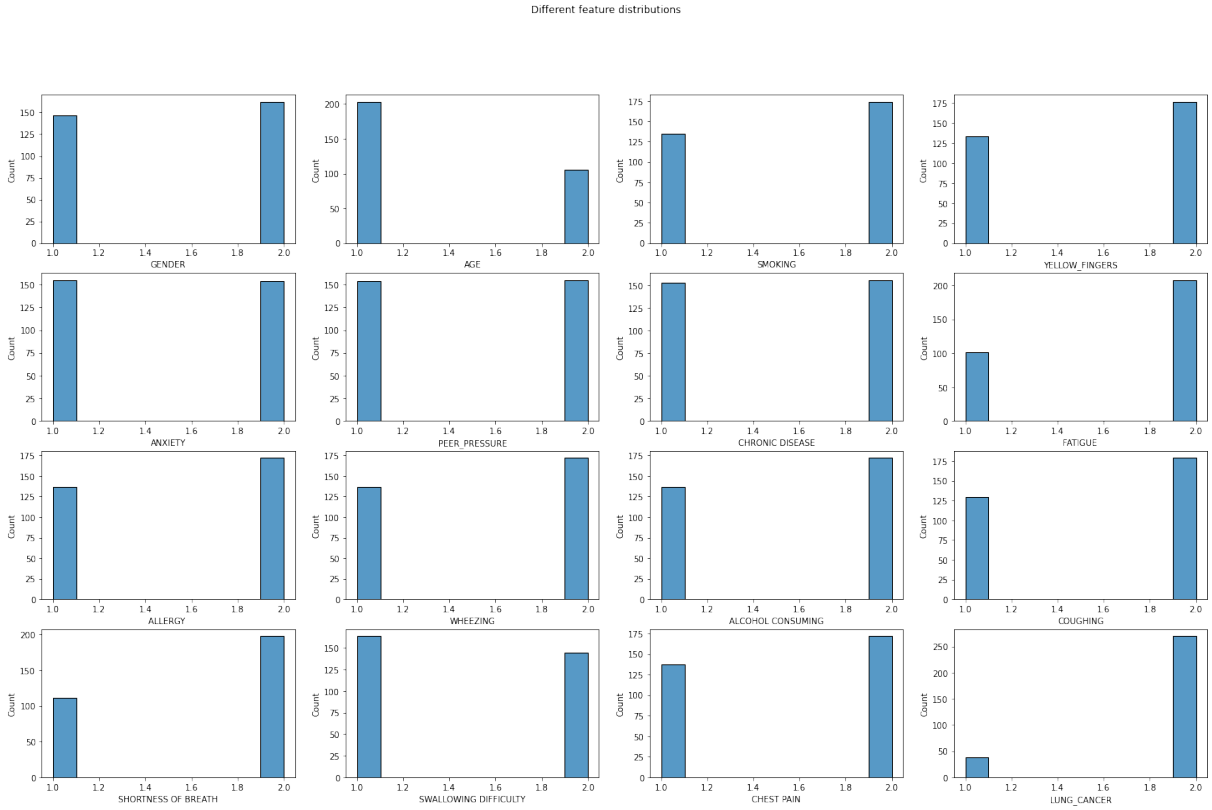


Rysunek 7: Sieć dla prognozy pogody

Zbiór był bardzo liczny (142193 próbek). Pozwoliło to na przetestowanie sieci na 10000 przykładach. Osiągnięta dokładność to 80.79%. Jest to bardzo zadowalająca wartość zważywszy, że model ten działa bardzo szybko.

5.3 Diagnoza raka płuc

Najbardziej zaawansowany zbiór poddany przez nas eksperymentom. Tutaj również będzie miało miejsce porównanie do innych metod klasyfikacji. Do dyspozycji mieliśmy 14 cech kategorycznych.



Rysunek 8: Rozkład danych

Niewątpliwą wadą tego zbioru są dwa aspekty: mała ilość próbek (309) oraz znaczna przewaga ilościowa próbek zakwalifikowanych jako chorych na raka, względem zdrowych.

5.3.1 Porównanie z innymi modelami

Dane podzieliliśmy na zbiór treningowy i testowy. Zbiór treningowy był wielkości 80% danych i zawierał 248 próbek, a treningowy pozostałe 20% (61 próbek). Po przetrenowaniu sieci bayesowskiej, przetestowaliśmy ją pod względem miary dokładności. Na danych treningowych osiągnęliśmy dokładność **87%**, a na danych testowych **89%**.

Aby ocenić jakość naszego modelu porównaliśmy go z innymi popularnymi klasyfikatorami. Wyniki eksperymentu przedstawiono w tabeli poniżej:

Miary	BBN	Regresja logiczna	Drzewo decyzyjne	SVC	Las losowy	Regresja logiczna	Sieć neuronowa
Dokładność na danych treningowych	87.09%	93.15%	92.74%	94.76%	95.16%	94.76%	94.76%
Dokładność na danych testowych	88.52%	93.44%	88.52%	91.8%	88.52%	90.16%	88.52%

6 Podsumowanie i wnioski

W niniejszym artykule przedstawiliśmy zasady działania oraz przykłady zastosowania sieci Bayesian Belief Network. Jak widzimy z powyższych eksperymentów sieci bayesowskie są dobrą metodą do klasyfikacji i znajdowania zależności w danych. Z dużą skutecznością możemy dzięki nim przewidzieć przyszłe zdarzenia i zaklasyfikować próbki danych. Dzięki swojej grafowej strukturze są łatwo interpretowalne i zrozumiałe dla człowieka. Są szybkie w działaniu gdyż opierają się o prosty model statystyczny. Pomimo swej prostej budowy osiągają wyniki podobne do popularnych klasyfikatorów. Podejście statystyczne w opisywanym modelu wyróżnia się na tle pozostałych znanych metod, co stanowi interesującą część uczenia maszynowego.

Przydatność sieci BBN mocno zależy od posiadanego zbioru - najlepiej, żeby dane były kategoryczne, co oznacza, że w przypadku niektórych zbiorów trzeba modyfikować dane (u nas w prognozie pogody). Natomiast próbki reprezentujące objawy bardzo dobrze nadają się do predykcji poprzez bayesowską sieć.

Bibliografia

[1] S. Dobilas. *BBN: Bayesian Belief Networks*. 2021. URL: https://towardsdatascience.com/bbn-bayesian-belief-networks-how-to-build-them-effectively-in-python-6b7f93435bba (term. wiz. 12. 12. 2021).

[2] D. Heckerman. „A Tutorial on Learning With Bayesian Networks". W: *CoRR* abs/2002.00269 (2020). arXiv: 2002.00269. URL: https://arxiv.org/abs/2002.00269.

[3] numFOCUS. *Pandas*. 2021. URL: https://pandas.pydata.org (term. wiz. 12. 12. 2021).

[4] Wikipedia. *Bayes' theorem*. 2021. URL: https://en.wikipedia.org/wiki/Bayes%5C%27_theorem (term. wiz. 12. 12. 2021).

[5] Wikipedia. *Bayesian network*. 2021. URL: https://en.wikipedia.org/wiki/Bayesian_network (term. wiz. 12. 12. 2021).