Sung Mo Yang

1. Machine learning fundamentals
    a. This is a bad example of model generalization because the narrator, similar to a machine learning algorithm, is too trained on the correlation of size and prices of houses specifically on Long Island. When put into a new environment with a different housing market like Texas, the narrator's ability to generalize will be lacking. It will judge houses in Texas based on Long Island size:price criteria, which is
    b. This is a good example of model generalization because the narrator is“trained” on similarly applicable environments with a plethora of significant factors. The narrator is informed of the effects that several factors have on rent prices. Even though the current situation changes with the year and COVID, the narrator can still use the factors to determine rent prices in this new environment. Essentially, the narrator is trained to see how factors affect rent prices and how differences in those factors affect rent prices. Therefore, when those factors differ due to COVID and different years, the narrator can still determine rent prices.
    c. This is a poor example of model generalization because the scenario clearly portrays overfitting. The neural network in the scenario is too effective and too accurate, indicating the inability to generalize. It has become too specific on training data and will have difficulty adjusting to new, radically different testing data.

2. Conditional independence vs independence

a.

| fur \ tail | furry | rope-like |
|---|---|---|
| blue | $\frac{5}{50} = \frac{1}{10}$ | 0 |
| gray | $\frac{5}{50} = \frac{1}{10}$ | 0 |
| brown | 0 | $\frac{40}{50} = \frac{4}{5}$ |

b. Without knowing the type of animal, the features, “fur color” and “tail texture” are not independent.

| Independence test | Calculated Probabilities | Actual Probabilities |
|---|---|---|
| P(blue and furry) = P(blue) * P(furry) | $\frac{5}{50} * \frac{10}{50} = \frac{50}{2500} = \frac{1}{50}$ | $\frac{1}{10}$ |
| P(blue and rope-like) = P(blue) * P(rope-like) | $\frac{5}{50} * \frac{40}{50} = \frac{200}{2500} = \frac{4}{50}$ | 0 |
| P(gray and furry) = P(gray) * P(furry) | $\frac{5}{50} * \frac{10}{50} = \frac{50}{2500} = \frac{1}{50}$ | $\frac{1}{10}$ |
| P(gray and rope-like) = P(gray) * P( rope-like) | $\frac{5}{50} * \frac{40}{50} = \frac{200}{2500} = \frac{4}{50}$ | 0 |
| P(brown and furry) = P(brown) * P(furry) | $\frac{40}{50} * \frac{10}{50} = \frac{400}{2500} = \frac{4}{25}$ | 0 |
| P(brown and rope-like) = P(brown) * P(rope-like) | $\frac{40}{50} * \frac{40}{50} = \frac{1600}{2500} = \frac{16}{25}$ | $\frac{4}{5}$ |

c.

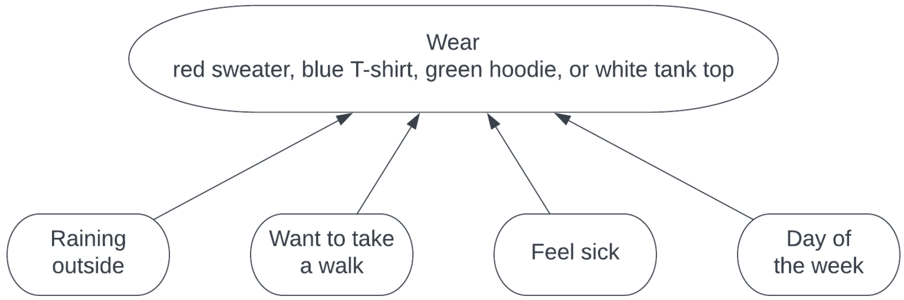
| fur \ tail | furry | rope-like |
|---|---|---|
| blue | $\frac{5}{10} = \frac{1}{2}$ | 0 |
| gray | $\frac{5}{10} = \frac{1}{2}$ | 0 |
| brown | 0 | 0 |

d. Knowing the animal is Tom's cherished baby daughter makes the features, "fur color" and "tail texture" independent.

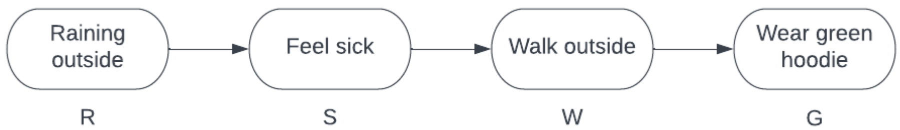
| Independence test | Calculated Probabilities | Actual Probabilities |
|---|---|---|
| P(blue and furry) = P(blue) * P(furry) | $\frac{5}{10} * \frac{10}{10} = \frac{50}{100} = \frac{1}{2}$ | $\frac{1}{2}$ |
| P(blue and rope-like) = P(blue) * P(rope-like) | $\frac{5}{10} * 0 = 0$ | 0 |
| P(gray and furry) = P(gray) * P(furry) | $\frac{5}{10} * \frac{10}{10} = \frac{50}{100} = \frac{1}{2}$ | $\frac{1}{2}$ |
| P(gray and rope-like) = P(gray) * P( rope-like) | $0 * \frac{10}{10} = 0$ | 0 |
| P(brown and furry) = P(brown) * P(furry) | $\frac{5}{10} * 0 = 0$ | 0 |
| P(brown and rope-like) = P(brown) * P(rope-like) | $0 * 0 = 0$ | 0 |

3. Directed graphical models and probability inference

a.




b.




| P(G \| W): Probability that I wear the green hoodie based on whether or not I walk outside | W: Probability that I walk outside |
|---|---|
| 100% | 1 |
| 0% | 0 |

| P(W \| S): Probability that I walk based on whether or not I feel sick | S: Probability that I feel sick |
|---|---|
| 10% | 1 |
| 60% | 0 |

| P(S \| R): Probability that I feel sick based on whether or not it rains | R: Probability that it rains |
|---|---|
| 70% | 1 |
| 15% | 0 |

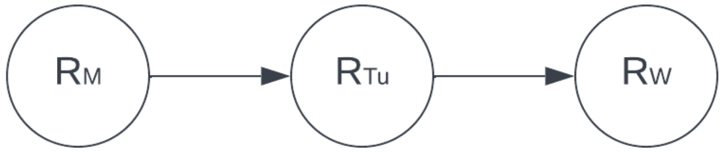
Given that it is raining, the probability that I am wearing a green hoodie:

$$P(G \mid R = 1) = \sum_{S, W \in \{0, 1\}} P(G \mid W)\, P(W \mid S)\, P(S \mid R)$$

| | | |
|---|---|---|
| W = 1, S = 1 | $P(G \mid W = 1)\, P(W = 1 \mid S = 1)\, P(S = 1 \mid R = 1)$ | 1 * 0.1 * 0.7 = 0.07 |
| W = 1, S = 0 | $P(G \mid W = 1)\, P(W = 1 \mid S = 0)\, P(S = 0 \mid R = 1)$ | 1 * 0.6 * 0.3 = 0.18 |
| W = 0, S = 1 | $P(G \mid W = 0)\, P(W = 0 \mid S = 1)\, P(S = 1 \mid R = 1)$ | 0 * 0.9 * 0.7 = 0 |
| W = 0, S = 0 | $P(G \mid W = 0)\, P(W = 0 \mid S = 0)\, P(S = 0 \mid R = 1)$ | 0 * 0.4 * 0.3 = 0 |

0.7 + 0.18 + 0 + 0 = 0.25

There is a 25% chance I am wearing a green hoodie




c.

Given: $P(T \mid R) \ = \ \{0.75 \, if \ R \ = \ 1, \ 0.25 \, if \ R \ = \ 0$

Given: $P(Rain\ today \mid Rain\ yesterday) \ = \ 0.7$

Given: $P(Rain\ today \mid Didn't\ rain\ yesterday) \ = \ 0.1$

$P(R_M = 1)$ = observed

$P(R_{Tu} = 1) = P(R_{Tu} = 1 \mid R_M = 1) = 0.7$

$P(R_W = 1) = P(R_W = 1 \mid R_{Tu} = 1)\ P(R_{Tu} = 1) + P(R_W = 1 \mid R_{Tu} = 0)\ P(R_{Tu} = 0)$

$P(R_W = 1) = 0.7 * 0.7 + 0.1 * 0.3$

$P(R_W = 1) = 0.52$

$P(T \mid R_w) = \sum_{R=\{0,1\}} P(T \mid R)\ P(R) \ = \ P(T \mid R \ = \ 0)\ P(R \ = \ 0) \ + \ P(T \mid R \ = \ 1)\ P(R \ = \ 1)$

$= \ 0.25 \ * \ 0.48 \ + \ 0.75 \ * \ 0.52 \ = \ 0.51$

The probability that I will wear a tank top on wednesday is 0.51

4. Naive Bayes and Alice in Wonderland

    a. Over the entire corpus, the classification accuracy was 0.250009874007622

```
def classification_accuracy():
    totalCount = 0
    accuracyCount = 0
    for i in range(0, len(corpus)-1):
        currentWord = corpus[i]
        mostLikeyNextWord = pred_2gram(currentWord)[0]

        print(i, mostLikeyNextWord, corpus[i+1])

        totalCount+=1
        if mostLikeyNextWord == corpus[i+1]:
            accuracyCount+=1

    return accuracyCount / totalCount

print(classification_accuracy())
print("done")
```

```
25302 little pleasure
25303 in in
25304 a all
25305 the their
25306 slates simple
25307 and joys
25308 remembering remembering
25309 her her
25310 head own
25311 business child
25312 said and
25313 the the
25314 queen happy
25315 summer summer
25316 day days
25317 the the
25318 queen end
0.2500098740076622
done
```

    b. For when n = 3, 5, 10

```
def classification_accuracy_n(nWords):
    totalCount = 0
    accuracyCount = 0
    for i in range(D-nWords):
        mostLikeyNextWord = pred_ngram([corpus[i+j] for j in range(nWords)])[0]
        if mostLikeyNextWord == corpus[i+nWords]:
            accuracyCount+=1
        totalCount+=1
    return accuracyCount / totalCount

print("n = 3: ", classification_accuracy_n(3))
print("n = 5: ", classification_accuracy_n(5))
print("n = 10: ", classification_accuracy_n(10))
```

```
n = 3:  0.550420665955682
n = 5:  0.745684376851669
n = 10:  0.9441722639273015
```

c. Text generation based on most likely next word

['it', 'said', 'the', 'king', 'and', 'the', 'queen', 'of', 'the', 'queen', 'and', 'the', 'queen', 'of', 'the', 'queen', 'and', 'the', 'queen', 'of', 'the', 'queen', 'and', 'the', 'queen']

d. Text generation based on sampling according to probability

['all', 'this', 'a', 'little', 'alice', 'in', 'the', 'little', 'of', 'the', 'right', 'said', 'the', 'queen', 'and', 'in', 'it', 'to', 'the', 'first', 'she', 'was', 'the', 'cat', 'and']

['all', 'this', 'a', 'little', 'alice', 'in', 'the', 'little', 'of', 'the', 'right', 'said', 'the', 'queen', 'and', 'in', 'it', 'to', 'the', 'first', 'she', 'was', 'the', 'cat', 'and']