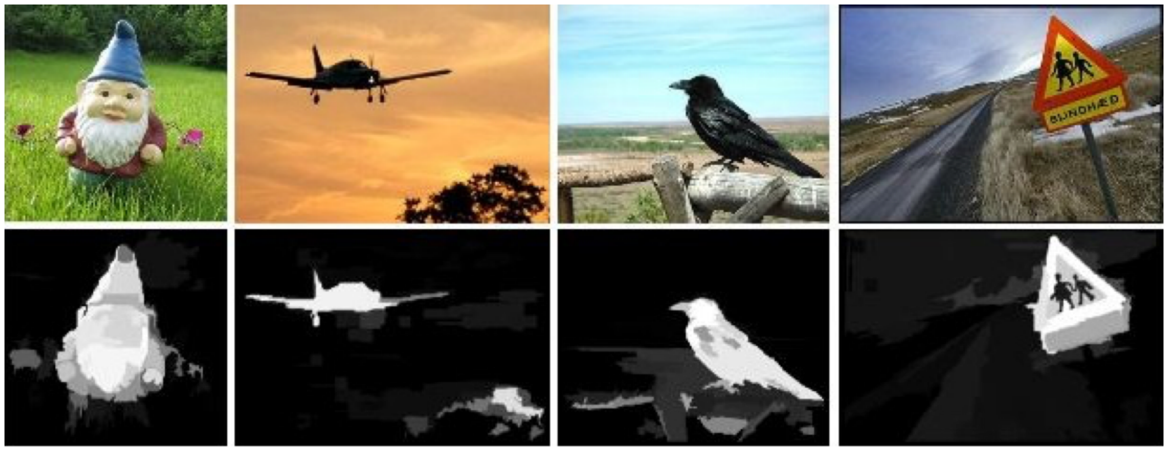
Chris Piech
CS 109

Section #9
Mar 12, 2025

Section 9

1 The Most Important Features (17 points)

Let's explore saliency, a measure of how important a feature is for classification. We define the saliency of the ith input feature for a given example $(\mathbf{x}, y)$ to be the absolute value of the partial derivative of the log likelihood, with respect to that input feature $\left|\frac{\partial LL}{\partial x_i}\right|$. Below, we show both input images and the corresponding saliency of their features (in this case, pixels) for an image classification model:



Consider a trained logistic regression classifier with weights θ that predicts binary class labels, y. In this question, we allow the values of $\mathbf{x}$ to be real numbers, which doesn't change the algorithm at all (neither training nor testing).

a. (4 points) What is the Log Likelihood of a single training example $(\mathbf{x}, y)$ for this logistic regression classifier?

b. (8 points) Calculate the saliency of a single feature (x_i) for one training example ($\mathbf{x}, y$).

c. (5 points) Show that the ratio of saliency for features i and j is the ratio of the absolute value of their weights $\frac{|\theta_i|}{|\theta_j|}$.

2 Timing Attack (23 points)

In this problem we will see how to crack a password in linear time by measuring how long the password check takes to execute (see code below).

```
# An insecure string comparison
def does_password_match(guess, password):
   n_guess = len(guess)
   n_password = len(password)
   if n_guess != n_password:
      return False                 # 4 lines executed to get here
   for i in range(n_guess):
      if guess[i] != password[i]:
         return False              # 6 + 2i lines executed to get here
   return True                     # 5 + 2n lines executed to get here
```

Assume that our server takes T ms to execute any line in the code where $T \sim N(\mu = 5, \sigma^2 = 0.5)$ milliseconds. The amount of time taken to execute a line is always independent of other lines.

On our site, all passwords only use lower case letters and are between 5 and 10 letters long, inclusive. A hacker is trying to crack the root password which is "gobayes" by carefully measuring how long the code takes to tell her that her guesses are incorrect.

a. (5 points) What is the distribution for the time it takes to execute k lines of code?

b. (7 points) First, the hacker needs to find the length of the password. What is the probability that the time taken to check a guess of correct length (when the server executes 6 lines) is longer than the time taken to check a guess of an incorrect length (when the server only executes 4 lines)? Assume the first letter of the guess does not match the password's first letter. Hint: $P(A > B) = P(A - B > 0)$.

c. (8 points) Now that our hacker knows the length of the password, to get the actual string, she will try to determine one letter at a time, starting with the first letter. To start, the hacker tries the string "aaaaaaa" and sees that it takes 27ms. Based on this timing, how much more probable is it that first character did not match (server executes 6 lines) than the first character did match (server executes 8 lines)? Assume that all letters in the alphabet are equally likely to be the first letter.

d. (3 points) If it takes the hacker 6 guesses to find the length of the password, and 26 guesses per letter to crack the password string, how many attempts does she need to crack our password, "gobayes"? Yikes!

3 Bayesian Carbon Dating (34 points)

We are able to know the age of ancient artifacts using a process called carbon dating. This process involves a lot of uncertainty! Living things have a constant proportion of a molecule called C14 in them. When living things die those molecules start to decay. The time to decay in years, T, of a C14 molecule is distributed as an exponential. $T \sim \text{Exp}(\lambda = 1/8267)$.

a. (5 points) Consider a single C14 molecule. What is the probability that it decays within 500 years?

b. (8 points) C14 molecules decay independently. A particular sample started with 100 molecules. What is the probability that exactly 95 are left after 500 years? Let p be your answer to part a.

c. (6 points) Write pseudocode for a function `pr_measure_given_age(m, age)` which returns $P(M = \texttt{m} | A = \texttt{age})$, the probability that exactly `m` molecules are left out of the original 100 after exactly `age` number of years.

d. (15 points) You observe 95 C14 molecules in a sample. You assume that the sample originally had 100 C14 molecules when it died. Write pseudocode for a function `age_belief()` that returns the full probability distribution $P(A = i | M = 95)$, where $A = i$ is the event that the sample died i years ago (here age is a discrete random variable). Use the function `pr_measure_given_age(m, age)` from part c. Your prior belief is that the sample's age was between $A = 500$ and $A = 600$ inclusive and that every year in that range is equally likely.