Handling Overfitting

- **Overfitting** occurs when the system finds regularities in the training set that are not in the test set.
- Often results in **overconfidence** (more extreme probabilities) and **overly complex models**.
Handling Overfitting

- **Overfitting** occurs when the system finds regularities in the training set that are not in the test set.
- Often results in *overconfidence* (more extreme probabilities) and *overly complex models*.
- Prefer simpler models. How do we trade off simplicity and fit to data?
- Test it on some hold-out data.
Bayes Rule:

\[ P(h|d) \propto P(d|h)P(h) \]

\[ \arg \max_h P(h|d) = \arg \max_h P(d|h)P(h) \]

\[ = \arg \max_h (\log P(d|h) + \log P(h)) \]

\[ \log P(d|h) \text{ measures fit to data} \]

\[ \log P(h) \text{ measures model complexity} \]
Bayes Rule:

\[ P(h|d) \propto P(d|h)P(h) \]

\[ \arg \max_h P(h|d) = \arg \max_h P(d|h)P(h) \]
\[ = \arg \max_h (\log P(d|h) + \log P(h)) \]

- \( \log P(d|h) \) measures fit to data
- \( \log P(h) \) measures model complexity
Regularization

Logistic regression, minimize sum-of-squares:

\[
\text{minimize } Error_E(w) = \sum_{e \in E} \left( Y(e) - f\left( \sum_{i} w_i X_i(e) \right) \right)^2.
\]
Regularization

Logistic regression, minimize sum-of-squares:

$$\text{minimize } Error_E(\overline{w}) = \sum_{e \in E} \left( Y(e) - f(\sum_i w_i X_i(e)) \right)^2.$$ 

L2 regularization (penalize deviation from $m$):

$$\text{minimize } Error_E(\overline{w}) + \lambda \sum_i (w_i - m)^2.$$ 

$L2$ regularization (penalize deviation from $m$):
Regularization

Logistic regression, minimize sum-of-squares:

$$\text{minimize } Error_E(\mathbf{w}) = \sum_{e \in E} \left( Y(e) - f\left( \sum_i w_i X_i(e) \right) \right)^2.$$ 

L2 regularization (penalize deviation from $m$):

$$\text{minimize } \sum_{e \in E} \left( Y(e) - f\left( \sum_i w_i X_i(e) \right) \right)^2 + \lambda \sum_i (w_i - m)^2$$

$\lambda$ is a parameter given a priori and/or learned.
Regularization

Logistic regression, minimize sum-of-squares:

\[
\text{minimize } Error_E(\overline{w}) = \sum_{e \in E} \left( Y(e) - f\left( \sum_i w_i X_i(e) \right) \right)^2.
\]

L2 regularization (penalize deviation from \(m\)):

\[
\text{minimize } \sum_{e \in E} \left( Y(e) - f\left( \sum_i w_i X_i(e) \right) \right)^2 + \lambda \sum_i (w_i - m)^2
\]

L1 regularization (penalize deviation from \(m\)):
Regularization

Logistic regression, minimize sum-of-squares:

$$\text{minimize } Error_E(\overline{w}) = \sum_{e \in E} \left( Y(e) - f\left( \sum_i w_i X_i(e) \right) \right)^2.$$ 

L2 regularization (penalize deviation from $m$):

$$\text{minimize } \sum_{e \in E} \left( Y(e) - f\left( \sum_i w_i X_i(e) \right) \right)^2 + \lambda \sum_i (w_i - m)^2$$

L1 regularization (penalize deviation from $m$):

$$\text{minimize } \sum_{e \in E} \left( Y(e) - f\left( \sum_i w_i X_i(e) \right) \right)^2 + \lambda \sum_i |w_i - m|$$

$\lambda$ is a parameter given a priori and/or learned.
L2 Regularization

Simplest case, no inputs: find \( p \) to minimize:

\[
\sum_i (p - d_i)^2 + \lambda (p - m)^2
\]
L2 Regularization

Simplest case, no inputs: find $p$ to minimize:

$$\sum_i (p - d_i)^2 + \lambda (p - m)^2$$

This is ambiguous! Why?
L2 Regularization

- Simplest case, no inputs: find \( p \) to minimize:

\[
\sum_i (p - d_i)^2 + \lambda (p - m)^2
\]

This is ambiguous! Why?

- Does it mean:

\[
0 \left( \sum_i (p - d_i)^2 \right) + \lambda (p - m)^2
\]

\[
1 \sum_i \left( (p - d_i)^2 + \lambda (p - m)^2 \right)
\]

- Does it matter?
Minimize:

\[
\left( \sum_i (p - d_i)^2 \right) + \lambda (p - m)^2
\]

Is at a minimum when:

\[
p = m + \lambda \sum_i d_i
\]

This is equivalent to a pseudocount with \(\lambda\) extra examples, each with value \(m\).
L2 Regularization: version 0

Minimize:

$$\left( \sum_i (p - d_i)^2 \right) + \lambda (p - m)^2$$

- Is at a minimum when:

  $$p = \frac{m\lambda + \sum_i d_i}{\lambda + n}$$

- This is equivalent to
Minimize:

\[
\left( \sum_i (p - d_i)^2 \right) + \lambda (p - m)^2
\]

- Is at a minimum when:

\[
p = \frac{m \lambda + \sum_i d_i}{\lambda + n}
\]

- This is equivalent to a pseudocount with \( \lambda \) extra examples, each with value \( m \).
L2 Regularization: version 1

Minimize:

$$\sum_i \left( (p - d_i)^2 + \lambda (p - m)^2 \right)$$

Is at a minimum when:

- Is at a minimum when:
Minimize:

$$\sum_i \left( (p - d_i)^2 + \lambda (p - m)^2 \right)$$

- Is at a minimum when:

  $$p = \frac{\lambda}{1 + \lambda} m + \frac{1}{1 + \lambda} \frac{\sum_i d_i}{n}$$

- This is equivalent to
L2 Regularization: version 1

Minimize:

\[ \sum_i \left( (p - d_i)^2 + \lambda (p - m)^2 \right) \]

- Is at a minimum when:

\[
p = \frac{\lambda}{1 + \lambda} m + \frac{1}{1 + \lambda} \frac{\sum_i d_i}{n}
\]

- This is equivalent to probabilistic mixture of \( m \) and the average of the data.
Gradient descent:

\[\text{procedure } \text{Learn}0(D, m, \eta, \lambda)\]
\[p \leftarrow m\]
\[\text{repeat}\]
\[\quad \text{for each } d_i \in D \text{ do}\]
Gradient descent:

procedure Learn0(D, m, η, λ)
    \( p \leftarrow m \)
    repeat
        for each \( d_i \in D \) do
            \( p \leftarrow p - \eta \times (p - d_i) \)
        \( p \leftarrow p - \eta \times \lambda \times (p - m) \)
    until termination
    return \( p \)
Gradient descent:

**procedure** $\text{Learn0}(D, m, \eta, \lambda)$

$p \leftarrow m$

repeat

for each $d_i \in D$ do

$p \leftarrow p - \eta \times (p - d_i)$

$p \leftarrow p - \eta \times \lambda \times (p - m)$

until termination

return $p$

**procedure** $\text{Learn1}(D, m, \eta, \lambda)$

$p \leftarrow m$

repeat

for each $d_i \in D$ do

$p \leftarrow p - \eta \times (p - d_i)$

$p \leftarrow p - \eta \times \lambda \times (p - m)$

until termination

return $p$
L2 regularization: issues

- Can’t we just distribute the $\lambda$ term out of the sum (as it doesn’t depend on $i$)?
L2 regularization: issues

- Can’t we just distribute the $\lambda$ term out of the sum (as it doesn’t depend on $i$)?
- How does the amount of data affect the prediction? (What if there is lots of data? What if there is very little?)
- How would the $\lambda$s be different?
- When should we use either one?
- Can we use both?
- How does it differ for minimizing log loss (maximizing log likelihood)?
- Is there a similar analysis for L1 regularization?
L2 regularization: issues

- Can’t we just distribute the $\lambda$ term out of the sum (as it doesn’t depend on $i$)?
- How does the amount of data affect the prediction? (What if there is lots of data? What if there is very little?)
- How would the $\lambda$s be different?
L2 regularization: issues

- Can’t we just distribute the $\lambda$ term out of the sum (as it doesn’t depend on $i$)?
- How does the amount of data affect the prediction? (What if there is lots of data? What if there is very little?)
- How would the $\lambda$s be different?
- When should we use either one?
- Can we use both? How does it differ for minimizing log loss (maximizing log likelihood)?
- Is there a similar analysis for L1 regularization?
L2 regularization: issues

- Can’t we just distribute the $\lambda$ term out of the sum (as it doesn’t depend on $i$)?
- How does the amount of data affect the prediction? (What if there is lots of data? What if there is very little?)
- How would the $\lambda$s be different?
- When should we use either one?
- Can we use both?
L2 regularization: issues

- Can’t we just distribute the $\lambda$ term out of the sum (as it doesn’t depend on $i$)?
- How does the amount of data affect the prediction? (What if there is lots of data? What if there is very little?)
- How would the $\lambda$s be different?
- When should we use either one?
- Can we use both?
- How does it differ for minimizing log loss (maximizing log likelihood)?
L2 regularization: issues

- Can’t we just distribute the $\lambda$ term out of the sum (as it doesn’t depend on $i$)?
- How does the amount of data affect the prediction? (What if there is lots of data? What if there is very little?)
- How would the $\lambda$s be different?
- When should we use either one?
- Can we use both?
- How does it differ for minimizing log loss (maximizing log likelihood)?
- Is there a similar analysis for L1 regularization?
Idea: split the training set into:
- new training set
- validation set

Use the new training set to train on. Use the model that works best on the validation set.

- To evaluate your algorithm, the test should not be used for training or validation.
- Many variants: k-fold cross validation, leave-one-out cross validation, etc.