Lecture 03:
Linear regression and regularization

Meelis Kull
meelis.kull@ut.ee
Fall 2018
✓ Tennis dataset
✓ Encoding categorical as numeric
✓ One-hot encoding
✓ Non-separable classes
✓ Maximum a posteriori (MAP) rule
✓ Bayes classifier
✓ Train and test data
✓ MAP with more features
✓ K nearest neighbours
✓ Naïve Bayes
Lecture 03 – Linear regression and regularization

• Main concepts in regression
• Linear regression
• Univariate ordinary least squares
• Multivariate ordinary least squares
• Regularization
• Ridge regression
• Lasso regression
Main concepts in regression

• Linear regression
• Univariate ordinary least squares
• Multivariate ordinary least squares
• Regularization
• Ridge regression
• Lasso regression
Regression

• Regression task is the same as classification task, except that we must predict a continuous variable (instead of a categorical class label)
Classifier vs Regression model

Instance → Classifier → Categorical Target (class label)

Instance → Regression model → Continuous Target (real number)
Mathematical notation of classification

• NB! Notations can vary across authors
• \( \mathcal{X} \) - input space (set of all possible instances)
• \( \mathcal{Y} \) - output space (all possible labels)
• \( f : \mathcal{X} \rightarrow \mathcal{Y} \) - any such function is a classifier
• \( x \in \mathcal{X} \) - instance
• \( y \in \mathcal{Y} \) - actual / true label of instance \( x \)
• \( \hat{y} = f(x) \) - predicted label of instance \( x \)
Mathematical notation of classification

- NB! Notations can vary across authors
- \( \mathcal{X} \) - input space (set of all possible instances)
- \( \mathcal{Y} \) - output space (all possible labels)
- \( f : \mathcal{X} \rightarrow \mathcal{Y} \) - any such function is a classifier
- \( \mathbf{x} \in \mathcal{X} \) - instance
- \( y \in \mathcal{Y} \) - actual / true label of instance \( \mathbf{x} \)
- \( \hat{y} = f(\mathbf{x}) \) - predicted label of instance \( \mathbf{x} \)

- Regression:
  - real numbers
  - regression model
  - target value
Simplified view on regression

• Suppose that there exists an actual / true function mapping the features to target variable $f^*: \mathcal{X} \rightarrow \mathbb{R}$

• In regression the task is to learn a function approximator $\hat{f}: \mathcal{X} \rightarrow \mathbb{R}$ such that $\hat{f} \approx f^*$

• For this we are given training data $(x_1, f^*(x_1)), (x_2, f^*(x_2)), \ldots, (x_n, f^*(x_n)) \in \mathcal{X} \times \mathbb{R}$
Simplified view on regression

\[ \mathcal{Y} = \mathbb{R} \]

\[ f^* : \mathcal{X} \rightarrow \mathbb{R} \]
Simplified view on regression

\[ \mathcal{Y} = \mathbb{R} \]

\[ f^* : \mathcal{X} \rightarrow \mathbb{R} \]

\[ f^* (x_1) \]

\[ x_1 \]
Simplified view on regression

\[ \mathcal{Y} = \mathbb{R} \]

\[ f^* : \mathcal{X} \rightarrow \mathbb{R} \]

\[ f^*(\mathbf{x}_1) \]

\[ f^*(\mathbf{x}_2) \]
Simplified view on regression

\[ Y = \mathbb{R} \]

\[ f^* : \mathcal{X} \rightarrow \mathbb{R} \]

\[ f^*(x_1) \]

\[ f^*(x_2) \]

\[ f^*(x_3) \]

\[ x_1 \]

\[ x_2 \]

\[ x_3 \]

\[ \mathcal{X} \]
Simplified view on regression

\[ \mathcal{Y} = \mathbb{R} \]

\[ f^*(\mathbf{x}_n) \]

\[ f^*(\mathbf{x}_2) \]

\[ f^*(\mathbf{x}_1) \]

\[ f^*(\mathbf{x}_3) \]

\[ f^*: \mathcal{X} \to \mathbb{R} \]
Simplified view on regression

\[ Y = \mathbb{R} \]

\[ f^*(x_n) \]

\[ f^*(x_2) \]

\[ f^*(x_1) \]

\[ f^*(x_3) \]

\[ f^*: \mathcal{X} \rightarrow \mathbb{R} \]

\[ \hat{f}: \mathcal{X} \rightarrow \mathbb{R} \]
Simplified view on regression

\[ \mathcal{Y} = \mathbb{R} \]

\[ f^* : \mathcal{X} \rightarrow \mathbb{R} \]

\[ f^* (x_n) \]

\[ f^* (x_2) \]

\[ f^* (x_1) \]

\[ f^* (x_3) \]

\[ \hat{f} : \mathcal{X} \rightarrow \mathbb{R} \]

\[ x_1 \]

\[ x_2 \]

\[ x_3 \]

\[ \ldots \]

\[ x_n \]
Simplified view?

• In what sense was this view simplified?

• Consider the following example task:
  – Predict the weight of a person from height
  – Suppose that there exists an actual / true function mapping the features to target variable

\[ f^* : \mathcal{X} \rightarrow \mathbb{R} \]
Simplified view?

• In what sense was this view simplified?
• Consider the following example task:
  – Predict the weight of a person from height
  – Suppose that there exists an actual / true function mapping the features to target variable
    \[ f^* : \mathcal{X} \rightarrow \mathbb{R} \]
  – Such function \( f^* \) does not exist! There are many people with same height but different weight
  – Height does not determine the weight uniquely
  – Relationship between height and weight is non-deterministic
Non-deterministic relationships

- In many regression tasks the features \( \mathbf{x} \) do not uniquely determine target variable \( y \)
- There does not exist one true function \( f^*: \mathcal{X} \rightarrow \mathbb{R} \) describing this relationship!
- Therefore, we cannot write training data as:
  \[(\mathbf{x}_1, f(\mathbf{x}_1)), (\mathbf{x}_2, f(\mathbf{x}_2)), \ldots, (\mathbf{x}_n, f(\mathbf{x}_n)) \in \mathcal{X} \times \mathbb{R}\]
- Instead, the training data are:
  \[(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n) \in \mathcal{X} \times \mathbb{R}\]
- Now that we don’t have \( f^* \), what do we learn?
What exactly do we want to learn?

• Do we want to learn \( \hat{f} \) such that:
  \[
  \hat{f}(x_1) \approx y_1, \hat{f}(x_2) \approx y_2, \ldots, \hat{f}(x_n) \approx y_n
  \]
  – No! This would mean good predictions on training data – relevant but not our main goal!

• We want to predict well on (future) test data!
  – On any future instance \( X \in \mathcal{X} \) with true (hidden) label \( Y \in \mathbb{R} \) we want \( \hat{f}(X) \approx Y \)

• What does it mean precisely?
Mean regression

• Suppose we want to predict a person’s weight $Y$ from height $X$.

• Ideally, we would like to estimate the probability distribution of weights for each given height: $p(Y | X)$.

• However, a regression model can predict just one real value.

• In **mean regression** (the usual type of regression) the task is taken to predict the mean: $f^*(X) = \mathbb{E}[Y | X]$. 
Mean regression

• In **mean regression** (the usual type of regression) the task is taken to predict the mean:

\[ f^* (X) = \mathbb{E} [ Y | X ] \]

• Hence, for any \( X \) we want to estimate the average expected value of the corresponding \( Y \)

• In our example:
  – Predict the average expected weight of people with a given height
Mean regression

\[ Y = \mathbb{R} \]
Mean regression

\( \mathcal{Y} = \mathbb{R} \)

\( f^* (X) = \mathbb{E} [Y | X] \)
Mean regression

\[ Y = \mathbb{R} \]

\[ f^*(X) = \mathbb{E}[Y \mid X] \]
Alternatives to mean regression

• Median regression:
  – Example: for a given height, predict the weight such that approximately half of the people with this height would be heavier than this.

• Quantile regression:
  – Median regression generalized to any other quantile (median is the 50%-quantile).
How to evaluate mean regression?

- How do we evaluate how well \( \hat{f} \) approximates \( f^*(X) = \mathbb{E}[Y | X] \)?
- Even if we have hold-out test data, we still do not know the true \( f^*(X) = \mathbb{E}[Y | X] \).
Evaluation of regression

• The mean $f^*(X) = \mathbb{E}[Y \mid X]$ is minimizing the expected **squared error** on future data

$$f^*(X) = \arg\min_{y \in \mathbb{R}} \mathbb{E}[(y - Y)^2 \mid X] = \mathbb{E}[Y \mid X]$$

• Therefore, the most usual evaluation measure in (mean) regression is **mean squared error (MSE)** on test data:

$$\frac{1}{|Te|} \sum_{(x, y) \in Te} (\hat{f}(x) - y)^2$$

– Because in expectation this is minimized at $\hat{f} = f^*$
Noise in regression

- **Noise** is the difference between the true label $Y$ and the mean $f^*(X) = \mathbb{E}[Y | X]$
  $$\varepsilon = Y - f^*(X)$$

- This relationship is usually presented as:
  $$Y = f^*(X) + \varepsilon$$

- The task in mean regression is to predict $f^*(X)$ and the noise cannot be (and should not attempted to be) predicted from the features $X$
Example on noise

• Suppose we want to predict a person’s weight $Y$ from height $X$

• What is the noise here?

• Noise is the difference between the actual weight and the average weight among people with the same height

• When trying to synthetically generate realistic data, then should apply mean regression and add noise
Notes about the definition of noise

• Some authors directly define the target as
  \[ Y = f^*(X) + \varepsilon \]
  and require noise to have zero mean

• The result is equivalent to our presentation, we just derived it in the opposite order
  – We defined the target first and then defined noise using target
More on evaluation of regression

- Two common measures:
  - MSE – mean squared error
    \[ MSE = \frac{1}{|Te|} \sum_{(x, y) \in Te} (\hat{f}(x) - y)^2 \]
  - RMSE – root mean squared error
    \[ RMSE = \sqrt{MSE} \]

- The advantage of RMSE: easier to interpret because it is measured in the same units as the target variable (whereas MSE is measured in these units squared)
We defined noise as:

A. $\varepsilon = Y - \mathbb{E}[Y \mid X]$
B. $\varepsilon = Y + f^*(X)$
C. $\varepsilon = Y - \hat{f}(X)$
D. $\varepsilon = X - f^*(Y)$
E. None of these
F. I don’t know
Some regression methods

- Linear regression
  - Ordinary least squares (OLS) regression
  - Ridge regression
  - Lasso regression
- Support vector regression (SVR)
- Regression trees
- Gradient boosting [machine] (GBM)
- Neural networks
- Gaussian processes (GP)
Main concepts in regression

- **Linear regression**
- Univariate ordinary least squares
- Multivariate ordinary least squares
- Regularization
- Ridge regression
- Lasso regression
Linear regression

• In regression the task is to learn a function approximator: \( \hat{f} : \mathcal{X} \rightarrow \mathbb{R} \)

• In linear regression:
  – We assume that the features are all numeric:
    \[ \mathcal{X} = \mathbb{R}^d \]
  – We must learn a linear model:
    \[ \hat{f}(\mathbf{x}) = \mathbf{w}_0 + \mathbf{w} \cdot \mathbf{x} = \mathbf{w}_0 + \sum_{i=1}^{d} w_i x_i \]
  
  • \( \mathbf{w}_0 \) - intercept
  • \( \mathbf{w} \) - coefficients (coefficient vector)
Univariate linear regression

• Linear regression with a single feature:
  – \( \mathbf{x} = (x_1) \)
  – \( \mathbf{w} = (w_1) \)

• We must learn a univariate linear function:
  \[
  \hat{f}(\mathbf{x}) = w_0 + \mathbf{w} \cdot \mathbf{x} = w_0 + w_1 x_1
  \]
  • \( w_0 \) - intercept
  • \( w_1 \) - slope

• Training data: \((x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n) \in \mathbb{R}^2\)

• Task is to learn \( w_0 \) and \( w_1 \) such that \( \hat{f} \) minimizes future squared error
Main concepts in regression

Linear regression

- Univariate ordinary least squares
- Multivariate ordinary least squares
- Regularization
- Ridge regression
- Lasso regression
Univariate ordinary least squares

• Ordinary least squares (OLS) regression learns the weights by minimizing MSE on training data
Ordinary least squares regression

\[ \mathcal{Y} = \mathbb{R} \]

\[ \hat{f} : \mathcal{X} \rightarrow \mathbb{R} \]
Univariate ordinary least squares

• Ordinary least squares (OLS) regression learns the weights by minimizing MSE on training data:

\[ \hat{w}_0, \hat{w}_1 = \arg\min_{w_0, w_1} MSE(w_0, w_1) \]
Minimization basics – 1D

• Find the point where derivative is zero

• Equivalently, point where gradient is zero
  – gradient = vector in the direction of steepest slope with length equal to derivative in that direction
Minimization basics – 2D

• Find the point where the gradient is zero
Minimization basics – 2D

• Find the point where the gradient is zero
Taking the gradient

\[ 0 = \nabla \text{MSE}(w_0, w_1) = \left( \frac{\partial \text{MSE}(w_0, w_1)}{\partial w_0}, \frac{\partial \text{MSE}(w_0, w_1)}{\partial w_1} \right) \]

\[
\left\{ \begin{array}{l}
0 = \frac{\partial \text{MSE}(w_0, w_1)}{\partial w_0} \\
0 = \frac{\partial \text{MSE}(w_0, w_1)}{\partial w_1}
\end{array} \right.
\]
Partial derivative with respect to $w_0$

\[
0 = \frac{\partial \text{MSE}(w_0, w_1)}{\partial w_0} = \frac{\partial}{\partial w_0} \frac{1}{|\text{Tr}|} \sum_{(x, y) \in \text{Tr}} ((w_0 + w_1 x) - y)^2
\]

\[
= \frac{1}{|\text{Tr}|} \sum_{(x, y) \in \text{Tr}} \frac{\partial}{\partial w_0} ((w_0 + w_1 x) - y)^2
\]

\[
= \frac{1}{|\text{Tr}|} \sum_{(x, y) \in \text{Tr}} \frac{\partial}{\partial w_0} ((w_0 + w_1 x) - y)^2 \cdot \frac{\partial ((w_0 + w_1 x) - y)}{\partial w_0}
\]

\[
= 2((w_0 + w_1 x) - y) \cdot 1
\]

\[
0 = \frac{1}{|\text{Tr}|} \sum_{(x, y) \in \text{Tr}} 2((w_0 + w_1 x) - y)
\]

\[
\frac{1}{|\text{Tr}|} \sum_{(x, y) \in \text{Tr}} y = w_0 + w_1 \frac{1}{|\text{Tr}|} \sum_{(x, y) \in \text{Tr}} x
\]

\[
\Rightarrow w_0 = \bar{y} - w_1 \bar{x}
\]
Partial derivative with respect to $w_1$

\[ 0 = \frac{\partial \text{MSE}(w_0, w_1)}{\partial w_1} = \frac{\partial}{\partial w_1} \frac{1}{|\mathcal{T}|} \sum_{(x,y) \in \mathcal{T}} ((w_0 + w_1 x) - y)^2 \]

\[ = \frac{1}{|\mathcal{T}|} \sum_{(x,y) \in \mathcal{T}} \frac{\partial ((w_0 + w_1 x) - y)^2}{\partial w_1} \]

\[ = \frac{1}{|\mathcal{T}|} \sum_{(x,y) \in \mathcal{T}} \frac{\partial ((w_0 + w_1 x) - y)^2}{\partial ((w_0 + w_1 x) - y)} \cdot \frac{\partial ((w_0 + w_1 x) - y)}{\partial w_1} \]

\[ = 2((w_0 + w_1 x) - y) \cdot x \]

\[ 0 = \frac{1}{|\mathcal{T}|} \sum_{(x,y) \in \mathcal{T}} 2((w_0 + w_1 x) - y) \cdot x \]

\[ 0 = \frac{1}{|\mathcal{T}|} \sum_{(x,y) \in \mathcal{T}} (\bar{y} - w_1 \bar{x} + w_1 x - y) \cdot x \]
Partial derivative with respect to $w_1$

\[
0 = \frac{1}{|\text{Tr}|} \sum_{(x, y) \in \text{Tr}} (\bar{y} - w_1 \bar{x} + w_1 x - y) \cdot x
\]

\[
0 = \sum_{(x, y) \in \text{Tr}} (\bar{y} - w_1 \bar{x} + w_1 x - y) \cdot x
\]

\[
w_1 \sum_{(x, y) \in \text{Tr}} (x - \bar{x}) \cdot x = \sum_{(x, y) \in \text{Tr}} (y - \bar{y}) \cdot x
\]

\[
w_1 \left( \sum_{(x, y) \in \text{Tr}} (x - \bar{x}) \cdot x - \sum_{(x, y) \in \text{Tr}} (x - \bar{x}) \cdot \bar{x} \right) = \sum_{(x, y) \in \text{Tr}} (y - \bar{y}) \cdot x - \sum_{(x, y) \in \text{Tr}} (y - \bar{y}) \cdot \bar{x}
\]

\[
w_1 \sum_{(x, y) \in \text{Tr}} (x - \bar{x})^2 = \sum_{(x, y) \in \text{Tr}} (y - \bar{y})(x - \bar{x})
\]

\[
w_1 = \frac{\sum_{(x, y) \in \text{Tr}} (y - \bar{y})(x - \bar{x})}{\sum_{(x, y) \in \text{Tr}} (x - \bar{x})^2} = \frac{\text{Cov}_{\text{Tr}}(x, y)}{\text{Var}_{\text{Tr}}(x)}
\]
Taking the gradient

\[ 0 = \nabla MSE(w_0, w_1) = \left( \frac{\partial MSE(w_0, w_1)}{\partial w_0}, \frac{\partial MSE(w_0, w_1)}{\partial w_1} \right) \]

\[ \begin{align*}
0 &= \frac{\partial MSE(w_0, w_1)}{\partial w_0} \\
0 &= \frac{\partial MSE(w_0, w_1)}{\partial w_1}
\end{align*} \]

Solution:

\[ \begin{align*}
w_1 &= \frac{Cov_{Tr}(x,y)}{Var_{Tr}(x)} \\
w_0 &= \overline{y} - w_1 \overline{x}
\end{align*} \]
Univariate ordinary least squares

- Ordinary least squares (OLS) regression learns the weights by minimizing MSE on training data:

\[ \hat{w}_0, \hat{w}_1 = \arg \min_{w_0, w_1} MSE(w_0, w_1) \]

\[ = \begin{cases} 
\hat{w}_1 = \frac{\text{Cov}_{Tr}(x, y)}{\text{Var}_{Tr}(x)} \\
\hat{w}_0 = \bar{y} - \hat{w}_1 \bar{x} 
\end{cases} \]
Effect of standardisation

• If before learning the regression model we standardise both the feature and the target variable (zero mean and unit variance)

• Then

\[ \hat{w}_0, \hat{w}_1 = \arg\min_{w_0, w_1} MSE(w_0, w_1) \]

\[ = \begin{cases} 
\hat{w}_1 = \frac{Cov_{Tr}(x, y)}{Var_{Tr}(x)} = Cov_{Tr}(x, y) = corr(x, y) \\
\hat{w}_0 = \bar{y} - \hat{w}_1 \bar{x} = 0 - \hat{w}_1 \cdot 0 = 0 
\end{cases} \]
Alternative interpretation of univariate OLS

- Univariate OLS can be interpreted as follows

  **Fitting:**
  - Standardising the feature and the target variable
  - Calculating the correlation between them

  **Predicting:**
  - Standardising the feature
  - Multiply it by the fitted correlation
  - Unstandardise the result (using the reverse of the operation applied when normalising the target variable)
OLS is very sensitive to outliers

- A single faraway point can significantly shift the predictions
OLS is very sensitive to outliers

- A single faraway point can significantly shift the predictions
Ordinary least squares minimizes

A. MSE on test data
B. MSE on train data
C. Error rate on train data
D. Error rate on test data
E. None of these
F. I don’t know
Lecture 03 – Linear regression and regularization

✓ Main concepts in regression
✓ Linear regression
✓ Univariate ordinary least squares
  • Multivariate ordinary least squares
  • Regularization
  • Ridge regression
  • Lasso regression
Multivariate ordinary least squares

- Ordinary least squares (OLS) regression learns the weights by minimizing MSE on training data:

\[
\hat{w}_0, \hat{w} = \arg\min_{w_0, w} \text{MSE}(w_0, w)
\]

\[
= \arg\min_{w_0, w} \frac{1}{|\text{Tr}|} \sum_{(x, y) \in \text{Tr}} ((w_0 + \sum_{i=1}^{d} w_i x_i) - y)^2
\]
Homogeneous coordinates

- Linear model
  \[ \hat{f}(x) = w_0 + w \cdot x = w_0 + \sum_{i=1}^{d} w_i x_i \]
- Let’s add a feature which is constantly \(= 1\)
  \[ x = (x_1, \ldots, x_d) \quad \rightarrow \quad x^\circ = (1, x_1, \ldots, x_d) = (x_0, x_1, \ldots, x_d) \]
  - These are called **homogeneous coordinates**
- Now we can rewrite the linear model:
  \[ \hat{f}(x^\circ) = w^\circ \cdot x^\circ = \sum_{i=0}^{d} w_i x_i \]
Multivariate OLS in homogeneous coordinates

• Rewriting multivariate OLS to use homogeneous coordinates:

\[ \hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \text{MSE}(\mathbf{w}) \]

\[ = \arg\min_{\mathbf{w}} \frac{1}{|\text{Tr}|} \sum_{(x^\circ, y) \in \text{Tr}} (x^\circ \cdot \mathbf{w}^\circ - y)^2 \]

• For notational simplicity we drop the \( \circ \) and always assume homogeneous coordinates
Rewriting multivariate OLS

- Denote the training instances by
  \[(x_1, y_1), \ldots, (x_n, y_n) \in \mathbb{R}^{d+1} \times \mathbb{R}\]
- Denote the residual errors on instances by:
  \[e_1 = y_1 - x_1 \cdot w, \ldots, e_n = y_n - x_n \cdot w\]
- In matrix form: \[e = y - Xw\]

  where:

  \[
  e = \begin{pmatrix}
    e_1 \\
    \vdots \\
    e_n
  \end{pmatrix},
  y = \begin{pmatrix}
    y_1 \\
    \vdots \\
    y_n
  \end{pmatrix},
  X = \begin{pmatrix}
    x_1 \\
    \vdots \\
    x_n
  \end{pmatrix},
  w = \begin{pmatrix}
    w_0 \\
    \vdots \\
    w_d
  \end{pmatrix}
  \]
Objective of OLS in matrix form

• We have residuals \( e = y - Xw \)

• We want to minimize mean squared error or equivalently the sum of squared errors:

\[
\hat{w} = \arg\min_w \sum_{i=1}^n e_i^2 = \arg\min_w e \cdot e = \arg\min_w (y - Xw) \cdot (y - Xw)
\]

• To solve this we equate the gradient to zero:

\[
0 = \nabla \left( (y - Xw) \cdot (y - Xw) \right)
\]

• Hence, the partial derivatives must be zero:

\[
0 = \frac{\partial \left( (y - Xw) \cdot (y - Xw) \right)}{\partial w}
\]
Calculating the partial derivatives

- Based on properties of matrix derivatives:

\[
0 = \frac{\partial ((y - Xw) \cdot (y - Xw))}{\partial w} = \frac{\partial ((y - Xw)^T (y - Xw))}{\partial w} = -2X^T (y - Xw)
\]

- From this:

\[
X^T Xw = X^T y
\]

\[
(X^T X)^{-1} (X^T X)w = (X^T X)^{-1} (X^T y)
\]

\[
w = (X^T X)^{-1} (X^T y)
\]
Multivariate OLS

• For multivariate OLS there exists a **closed-form solution** (i.e. can be explicitly calculated without numerical optimization):

\[ \hat{w} = \arg \min_w \left( (y - Xw) \cdot (y - Xw) \right) = (X^T X)^{-1} (X^T y) \]
Interpretation of multivariate OLS

• In the formula \( \hat{w} = (X^T X)^{-1} (X^T y) \)
  - \( (X^T X)^{-1} \) decorrelates, centers and normalizes the features
  - \( X^T y \) calculates the covariances (if features and target variable are centred)
Closed-form solutions exist for:

A. Univariate OLS
B. Multivariate OLS
C. Both of these
D. None of these
E. I don’t know
Pros and cons of OLS

• Pro: if the number of instances is much bigger than the number of features \((n \gg d)\) then works quite well

• Con: if not \(n \gg d\) then OLS tends to overfit the noise, particularly if there is a lot of noise

• Con: if many features are collinear (highly correlated) then OLS tends to overfit
Main concepts in regression
Linear regression
Univariate ordinary least squares
Multivariate ordinary least squares

- **Regularization**
- Ridge regression
- Lasso regression
Regularization

- The main idea of regularization in machine learning is to penalize complex models
- If two models have a similar loss on the training data then the simpler tends to be better on test data
- Simpler models have less capacity to overfit the noise
How does OLS overfit noise?

• If the features contain only noise, then it is best to learn a constant mean prediction, with all coefficients zero except the intercept.
• OLS learns non-zero weights, fitting the noise as well as it can.
• The length of the test-optimal weight vector tends to indicate how predictable the target variable is (longer means more predictable).
• When overfitting, OLS tends to produce a longer weight vector than test-optimal.
Lecture 03 – Linear regression and regularization

✓ Main concepts in regression
✓ Linear regression
✓ Univariate ordinary least squares
✓ Multivariate ordinary least squares
✓ Regularization
  • Ridge regression
  • Lasso regression
Ridge regression

- Ridge regression is a regularisation method for linear regression
- It regularises the task by penalising vector length:
  \[
  \hat{w} = \arg\min_w \left( (y - Xw) \cdot (y - Xw) \right) + \lambda ||w||^2
  \]
- Here \( \lambda \) is a regularisation parameter
  - Higher \( \lambda \) means stronger regularization
  - If \( \lambda = 0 \) then we are back at OLS
What if we choose $\lambda$ by minimizing regularised MSE on training data?

A. Yes, this is a good strategy
B. No, this always gives $\lambda = 0$
C. No, this results with too big $\lambda$
D. None of these
E. I don’t know
How to choose $\lambda$?

- Try different $\lambda$ and evaluate which gives lowest MSE on a hold-out validation data (data that are not used in training the model)

- For better result use K-fold cross-validation:
  - Split training data into K folds
  - For $i=1,2,\ldots,K$
    - Choose $i$-th fold to be the validation fold and train the model on all other folds, measure MSE on the $i$-th fold
    - Average all above MSE separately for each $\lambda$
    - Choose $\lambda$ which gives the lowest average MSE
Train-test split

The whole dataset 100%
Train-test split

Training data 80%  Test 20%
4-fold cross-validation

Training data 80%
4-fold cross-validation

Training data 80%
4-fold cross-validation

Training data 80%

Train on 60% of data

Validate on 20%
4-fold cross-validation

Training data 80%

Train on 60% of data  Validate on 20%
4-fold cross-validation

Training data 80%

Train on 60% of data    Validate on 20%

0.75
4-fold cross-validation

Training data 80%

Train  Train  Train  Val  • ➞  0.75
Train  Train  Val   Train  • ➞  0.85
4-fold cross-validation

Training data 80%

- Train, Train, Train, Val → 0.75
- Train, Train, Val, Train → 0.85
- Train, Val, Train, Train → 0.91
4-fold cross-validation

Training data 80%

- 20% Train
- 20% Train
- 20% Train
- 20% Val

- 0.75
- 0.85
- 0.91
- 0.68
4-fold cross-validation

Training data 80%

MEAN (0.75, 0.85, 0.91, 0.68) = ?
4-fold cross-validation

Training data 80%

MEAN (0.75, 0.85, 0.91, 0.68) = 0.75
4-fold cross-validation

MEAN (0.75, 0.85, 0.91, 0.68) = 0.75

Choose the best model/parameters based on this estimate and then apply it to test set.
Ridge regression

- Ridge regression is a regularisation method for linear regression
- It regularises the task by penalising vector length:
  \[
  \hat{w} = \arg\min_w \left( (y - Xw) \cdot (y - Xw) \right) + \lambda ||w||^2
  \]
- Here \( \lambda \) is a regularisation parameter
  - Higher \( \lambda \) means stronger regularization
  - If \( \lambda = 0 \) then we are back at OLS
Calculating the partial derivatives

- Based on properties of matrix derivatives:

\[
0 = \frac{\partial ((y - Xw)^T (y - Xw) + \lambda w^T w)}{\partial w} = -2X^T (y - Xw) + 2\lambda w
\]

- From this:

\[
X^T Xw + \lambda w = X^T y
\]

\[
(X^T X + \lambda I)w = X^T y
\]

\[
(X^T X + \lambda I)^{-1} (X^T X + \lambda I)w = (X^T X + \lambda I)^{-1} X^T y
\]

\[
w = (X^T X + \lambda I)^{-1} X^T y
\]
Ridge regression

• For ridge regression there exists a **closed-form solution** (i.e. can be explicitly calculated without numerical optimization):

\[
\hat{w} = \arg\min_w \left( (y - Xw) \cdot (y - Xw) + \lambda ||w||^2 \right)
\]

\[
= (X^T X + \lambda I)^{-1} X^T y
\]
Name of ridge regression

• Ridge regression:
  – Mountain ridge – many points with same height, all local maxima
  – If inverting this intuition then many local minima in a valley
  – Ridge regression makes these into a single optimum

• Also known as:
  – Linear regression with Tikhonov regularisation
  – Linear regression with L2-norm regularisation
  – L2-regularised linear regression
Pros and cons of ridge regression

- **Pro**: effective in reducing overfitting compared to OLS
- **Con**: all coefficients are still non-zero, even if the true model is very sparse (very few non-zero coefficients)
Example of ridge regression

- Regularization path (coefficients, varying $\lambda$)

This is on an example dataset with 8 features (lcavol, svi, ..., lcp)

NB! X-axis here is not $\lambda$ but is an anti-monotonic function of $\lambda$ (degrees of freedom in regularisation)
Lecture 03 – Linear regression and regularization

✓ Main concepts in regression
✓ Linear regression
✓ Univariate ordinary least squares
✓ Multivariate ordinary least squares
✓ Regularization
✓ Ridge regression
• Lasso regression
Lasso regression

- Lasso regression is another regularisation method for linear regression
- It regularises the task by penalising the sum of absolute values of weights:
  \[
  \hat{w} = \arg \min_w \left( (y - Xw) \cdot (y - Xw) \right) + \lambda \sum_{i=1}^{n} |w_i|
  \]
- No closed-form solution exists, must be optimised numerically
Lasso regression:

- Lasso = Least Absolute Shrinkage and Selection Operator

Also known as:

- Linear regression with L1-norm regularisation
- L1-regularised linear regression
Lasso vs ridge regression

• Lasso can produce sparse solutions
  – Many coefficients are exactly zero
  – How many? Depends on \( \lambda \)

• This is beneficial for two reasons:
  – The learned model becomes easier to interpret
  – If the true model is sparse, then lasso tends to give lower test MSE than ridge regression

• The parameter \( \lambda \) can be learned by using a hold-out validation dataset or K-fold cross-validation
Example of lasso regression

- Regularization path (coefficients, varying $\lambda$)

This is on an example dataset with 8 features (lcavol, svi, ..., lcp)

NB! X-axis here is not $\lambda$ but is an anti-monotonic function of $\lambda$ (shrinkage factor in regularisation)
Pros and cons of lasso regression

• **Pro:** if the true model is sparse then it outperforms ridge regression and OLS
• **Con:** if several features are highly correlated then it tends to put high weight on an arbitrary one of those and zero to others
Lasso and ridge regression methods…

A. Learn linear models
B. Learn non-linear models
C. Lasso is linear, ridge non-linear
D. Ridge is linear, lasso non-linear
E. I don’t know
Mathematical notation of supervised learning

- NB! Notations can vary across authors
- \( \mathcal{X} \) - input space (set of all possible instances)
- \( \mathcal{Y} \) - output space (all possible target values)
- \( f : \mathcal{X} \rightarrow \mathcal{Y} \) - any such function is a predictive model
- \( x \in \mathcal{X} \) - instance
- \( y \in \mathcal{Y} \) - actual / true target value of instance \( x \)
- \( \hat{y} = f(x) \) - predicted target value of instance \( x \)
Main concepts in regression
Linear regression
Univariate ordinary least squares
Multivariate ordinary least squares
Regularization
Ridge regression
Lasso regression