Outline

• **Linear regression**
  • Auto data set

• **Logistic regression**
  • Auto data set
### Dataset Auto from the ISLR package

392 instances on the following 9 features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mpg</td>
<td>Miles per gallon</td>
</tr>
<tr>
<td>cylinders</td>
<td>Number of cylinders between 4 and 8</td>
</tr>
<tr>
<td>displacement</td>
<td>Engine displacement (cu. inches)</td>
</tr>
<tr>
<td>horsepower</td>
<td>Engine horsepower</td>
</tr>
<tr>
<td>weight</td>
<td>Vehicle weight (lbs.)</td>
</tr>
<tr>
<td>acceleration</td>
<td>Time to accelerate from 0 to 60 mph (sec.)</td>
</tr>
<tr>
<td>year</td>
<td>Model year (modulo 100)</td>
</tr>
<tr>
<td>name</td>
<td>Vehicle name</td>
</tr>
</tbody>
</table>
Dataset Auto from the ISLR package
Linear regression
Linear regression is a class of regression algorithms assuming that there is at least a linear dependence between a target attribute and features.

A target hypothesis $f$ has a form of **linear function**

$$f(x; \Theta) = \theta_0 + \theta_1 x_1 + \cdots + \theta_m x_m$$  \hspace{1cm} (1)

- $\theta_0, \ldots, \theta_m$ are regression parameters
- we think of them as weights that determine how each feature affects the prediction
- **simple linear regression** if $m = 1$
Linear regression

Notation

\[
y = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}, \quad \Theta^\top = \begin{pmatrix} \theta_0 \\ \vdots \\ \theta_m \end{pmatrix}, \quad X = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1m} \\ 1 & x_{21} & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{nm} \end{pmatrix}
\]

Now we can write \( y = X\Theta^\top \), \( f(x) = \Theta^\top x \)
Parameter interpretation

Numerical feature

$\theta_i$ is the average change in $y$ for a unit change in $A_i$ holding all other features fixed.
Categorical feature with $k$ values

- replace it with $k - 1$ dummy variables $DA^1, DA^2, \ldots, DA^{k-1}$

**Example:** run simple linear regression $mpg \sim origin$

<table>
<thead>
<tr>
<th></th>
<th>$DA^1$</th>
<th>$DA^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>American</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>European</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Japanase</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

- $y = \theta_0 + \theta_1 DA^1 + \theta_2 DA^1$
- $y = \theta_0 + \theta_1$ if the car is European
- $y = \theta_0 + \theta_2$ if the car is Japanese
- $y = \theta_0$ if the car is American
- $\theta_0$ as the average $mpg$ for American cars
- $\theta_1$ as the average difference in $mpg$ between European and American cars
- $\theta_2$ as the average difference in $mpg$ between Japanese and American cars
Parameter estimates
Least Square Method

- residual $y_i - \hat{y}_i$, where $\hat{y}_i = \hat{f}(x_i) = \hat{\Theta}^T x_i$
- **Loss function** Residual Sum of Squares $\text{RSS}(\hat{\Theta}) = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$
Parameter estimates
Least Square Method

Optimization problem

\[ \Theta^* = \arg\min_{\Theta} \text{RSS}(\Theta) \]

The \( \arg\min \) operator will give \( \Theta \) for which \( \text{RSS}(\Theta) \) is minimal.
Parameter estimates
Least Square Method

Solving the optimization problem analytically

Normal Equations Calculus

**Theorem**

Θ* is a least square solution to \( y = X\Theta^T \iff \Theta^* \) is a solution to the Normal equation \( X^T X\Theta = X^T y \).

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

Computational complexity of a \((m + 1) \times (m + 1)\) matrix inversion is \( O((m + 1)^3) \) :-(
Parameter estimates
Least Square Method

Solving the optimization problem numerically

Gradient Descent Algorithm
**Assume:** simple regression, $\theta_0 = 0$, $\theta_1 \neq 0$
**Gradient Descent Algorithm**

Assume: simple regression, $\theta_0 \neq 0$, $\theta_1 \neq 0$

Loss Function $L$ has a minimum value at the red point

Contours of Loss Function
Gradient Descent Algorithm

Gradient descent algorithm is an optimization algorithm to find a local minimum of a function $f$. 

\[ f(x) \]

\[ f(z) \]

\[ z \]
1. Start with some $x_0$. 

\begin{tikzpicture}
  \draw[->,thick] (-3,0) -- (3,0) node[right] {$x$};
  \draw[->,thick] (0,-3) -- (0,3) node[above] {$f(x)$};
  \draw[thick] plot[domain=-2:2,samples=50] ({\x},{\x^2});
  \filldraw[red] (0,0) circle (2pt) node[above right] {step 0};
  \draw[blue] (0,-2) -- (0,2) node[below] {$f(z)$};
  \draw[blue] (-2,0) -- (2,0) node[right] {$z$};
\end{tikzpicture}
2. Keep changing $x_i$ to reduce $f(x_i)$
Which direction to go? How big step to do?
Gradient Descent Algorithm

Credits: Andrew Ng

NPFL054, 2019 Hladká & Holub Lecture 4, page 19/50
Gradient Descent Algorithm

- We are seeking the solution to the minimum of a function $f(x)$. Given some initial value $x_0$, we can change its value in many directions.
- What is the best direction to minimize $f$? We take the gradient $\nabla f$ of $f$

$$\nabla f(x_1, x_2, \ldots, x_m) = \langle \frac{\partial f(x_1, x_2, \ldots, x_m)}{\partial x_1}, \ldots, \frac{\partial f(x_1, x_2, \ldots, x_m)}{\partial x_m} \rangle$$

- Intuitively, the gradient of $f$ at any point tells which direction is the steepest from that point and how steep it is. So we change $x$ in the opposite direction to lower the function value.
Gradient Descent Algorithm

Choice of the step: assume constant value

If the step is too small, GDA can be slow.
Choice of the step

If the step is too large, GDA can overshoot the minimum. It may fail to converge, or even diverge.
Gradient Descent Algorithm

repeat until convergence {

$$\Theta^{K+1} := \Theta^K - \alpha \nabla f(\Theta^K)$$

}

$- \alpha$ is a positive step-size hyperparameter

l.e. simultaneously update $\theta_j$, $j = 1, \ldots, m$
For linear regression $f = RSS$,

$$\theta_{j}^{K+1} := \theta_{j}^{K} - \alpha \frac{1}{n} \sum_{i=1}^{n} (f(x_i; \Theta^K) - y_i)x_{ij}$$

RSS is a convex function, so there is no local optimum, just global minimum.
**Polynomial regression** is an extension of linear regression where the relationship between features and target value is modelled as a $d$-th order polynomial.

**Simple regression**

$$y = \Theta_0 + \Theta_1 x_1$$

**Polynomial regression**

$$y = \Theta_0 + \Theta_1 x_1 + \Theta_2 x_1^2 + \ldots \Theta_d x_1^d$$

It is still a linear model with features $A_1, A_1^2, \ldots, A_1^d$.

The *linear* in linear model refers to the hypothesis parameters, not to the features. Thus, the parameters $\Theta_0, \Theta_1, \ldots, \Theta_d$ can be easily estimated using least squares linear regression.
Polynomial regression
Auto data set

ISLR: Auto data set

- Linear
- Degree 2
- Degree 5
Assessing the accuracy of the model

- **Coefficient of determination** $R^2$ measures the proportion of variance in a target value that is reduced by taking into account $x$

\[
R^2 = \frac{TSS - RSS}{TSS} = 1 - \frac{RSS}{TSS}
\]

where Total Sum of Squares $TSS = \sum_{i=1}^{n}(y_i - \bar{y})^2$; $R^2 \in (0, 1)$

- **Mean Squared Error** $MSE$

\[
MSE = \frac{1}{n} \cdot RSS
\]
Binary classification
Decision boundary

\[ Y = \{0, 1\} \]

Decision boundary takes a form of function \( f \) and partitions a feature space into two sets, one for each class.
**Hyperplane** is a linear decision boundary of the form

\[ \Theta^\top \mathbf{x} = 0 \]

where direction of \( \langle \theta_1, \theta_2, \ldots, \theta_m \rangle \) is perpendicular to the hyperplane and \( \theta_0 \) determines position of the hyperplane with respect to the origin.
Binary classification

Hyperplane

- point if $m = 1$, line if $m = 2$, plane if $m = 3$, ... 
- we can use hyperplane for classification so that 

$$f(x) = \begin{cases} 
1 & \text{if } \theta_0 + \theta_1 x_1 + \cdots + \theta_m x_m \geq 0 \\
0 & \text{if } \theta_0 + \theta_1 x_1 + \cdots + \theta_m x_m < 0 
\end{cases}$$

- **linear classifiers** classify examples using hyperplane
Binary classification
Can we use linear regression?

We are heading logistic regression.

(Yes) 1

(No) 0
Binary classification
Can we use linear regression?

Fit the data with a linear function $f$

$$f(x) = \Phi^T x$$

(Yes) 1

(No) 0
Binary classification
Can we use linear regression?

Classify

- if \( f(x) \geq 0.5 \), predict 1
- if \( f(x) < 0.5 \), predict 0
Binary classification
Can we use linear regression?

Add one more training instance

What to do if $f(x) > 1$ or $f(x) < 0$?
Logistic regression
Logistic regression is a classification algorithm.

Its target hypothesis $f$ for a binary classification has a form of sigmoid function:

$$f(x; \Theta) = \frac{1}{1 + e^{-\Theta^\top x}} = \frac{e^{\Theta^\top x}}{1 + e^{\Theta^\top x}}$$

- $f(z) = g(z) = \frac{1}{1 + e^{-z}}$
- $\lim_{z \to +\infty} g(z) = 1$
- $\lim_{z \to -\infty} g(z) = 0$
Classification rule

Predict a target value using $\hat{f}(x; \hat{\Theta})$ so that

- if $\hat{f}(x; \hat{\Theta}) \geq 0.5$, i.e. $\hat{\Theta}^T x \geq 0$, predict 1
- if $\hat{f}(x; \hat{\Theta}) < 0.5$, i.e. $\hat{\Theta}^T x < 0$, predict 0
Modeling conditional probabilities

Logistic regression models the conditional probability $Pr(y = 1|x; \Theta)$

$$f(x; \Theta) = Pr(y = 1|x; \Theta) = \frac{1}{1 + e^{-\Theta^T x}}$$

Algebraic manipulation results in

$$\frac{Pr(y = 1|x; \Theta)}{1 - Pr(y = 1|x; \Theta)} = e^{\Theta^T x} \in (0, +\infty)$$

Take logarithm

$$\ln \frac{f(x; \Theta)}{1 - f(x; \Theta)} = \Theta^T x \in (-\infty, +\infty)$$
Modeling conditional probabilities

- odds = Pr(y = 1|x; Θ)/ Pr(y = 0|x; Θ)
- log-odds = logit
- logit(p) = ln(p/(1-p))
- logit is linear in x
Parameter interpretation
Numerical features

$\theta_i$ gives an average change in $\text{logit}(f(x))$ with one-unit change in $A_i$ holding all other features fixed.
Parameter interpretation
Binary features

Example:

<table>
<thead>
<tr>
<th>disease</th>
<th>female 0 (male)</th>
<th>1 (female)</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>no</td>
<td>74</td>
<td>77</td>
<td>151</td>
</tr>
<tr>
<td>yes</td>
<td>17</td>
<td>32</td>
<td>49</td>
</tr>
<tr>
<td>Total</td>
<td>91</td>
<td>109</td>
<td>200</td>
</tr>
</tbody>
</table>

- the odds of having the disease for male:
  \[ \frac{\Pr(\text{disease} = \text{yes}|\text{female} = 0)}{\Pr(\text{disease} = \text{no}|\text{female} = 0)} = \frac{17}{74} / \frac{91}{91} = 0.23 \]

- the odds of having the disease for female:
  \[ \frac{\Pr(\text{disease} = \text{yes}|\text{female} = 1)}{\Pr(\text{disease} = \text{no}|\text{female} = 1)} = \frac{32}{109} / \frac{77}{109} = 0.42 \]

- the ratio of the odds for female to the odds for male \( 0.42 / 0.23 = 1.81 \), i.e. the odds for female are about 81% higher than the odds for males
Parameter interpretation
Binary features

\[
\ln \frac{p}{1-p} = \theta_0 + \theta_1 \ast \text{female}
\]

**If** female == 0

- \(p = p_1 \rightarrow \frac{p_1}{1-p_1} = e^{\theta_0}\)
- the intercept \(\theta_0\) is the log odds for men

**If** female == 1

- \(p = p_2 \rightarrow \frac{p_2}{1-p_2} = e^{\theta_0 + \theta_1}\)
- odds ratio = \(\frac{p_2}{1-p_2} / \frac{p_1}{1-p_1} = e^{\theta_1}\)
- the parameter \(e^{\theta_1}\) is the odds ratio between women and men

Assume the output of logistic regression \(\theta_0 = -1.471, \theta_1 = 0.593\). Then relate the odds for males and females and the parameters: \(-1.471 = \ln 0.23, 0.593 = \ln 1.81\)
Parameter estimates

- **Loss function**

\[
L(\Theta) = - \sum_{i=1}^{n} y_i \log P(y_i|x_i; \Theta) + (1 - y_i) \log (1 - P(y_i|x_i; \Theta))
\]

See Maximum Likelihood Principle for derivation of this loss function.

- **Optimization problem**

\[
\Theta^* = \arg\min_{\Theta} L(\Theta)
\]
Parameter estimates
Gradient Descent Algorithm

repeat until convergence {

\[ \Theta^{K+1} := \Theta^K - \alpha \nabla f(\Theta^K) \]

}

– \( \alpha \) is a positive step-size hyperparameter

i.e. simultaneously update \( \theta_j, j = 1, \ldots m \)

\[ \theta_j^{K+1} := \theta_j^K - \alpha \frac{1}{n} \sum_{i=1}^{n} \left( f(x_i; \Theta^K) - y_i \right) x_{ij} \]
Non-linear decision boundary

- Let \( f(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2) \) (a higher degree polynomial)
- Assume \( \theta_0 = -1, \theta_1 = 0, \theta_2 = 0, \theta_3 = 1, \theta_4 = 1 \)
- Predict \( y = 1 \) if \(-1 + x_1^2 + x_2^2 \geq 0\), i.e. \( x_1^2 + x_2^2 \geq 1 \)
Non-linear decision boundary
Logistic regression
Summary

Classification of $x$ by $\hat{f}^*$

1. Project $x$ onto $\hat{\Theta}^*$ to convert it into a real number $z$ in the range $\langle -\infty, +\infty \rangle$
   - i.e. $z = \hat{\Theta}^* \top x$
2. Map $z$ to the range $\langle 0, 1 \rangle$ using the sigmoid function $g(z) = 1/(1 + e^{-z})$
3. Classify $x$ using a classification rule
\(|Y| = N, N \geq 3\)

- **One-to-all**
  - train \(N\) predictors \(f_k\) for the pair \(k\)-th class and \(\{1, \cdots, N\} \setminus \{k\}\) classes
  - classify \(x\) into the class \(k^* = \text{argmax}_k f_k(x)\)

- **One-to-one**
  - train \(\binom{N}{2}\) classifiers \(f_i\)
  - classify \(x\) into the class \(k^* = \max_{k=1,\ldots,N} \sum_{i=1}^{\binom{N}{2}} \delta(f_i(x) = k)\)
Logistic regression
Multi-class classification

One-to-all
Summary of Examination Requirements

- Linear regression, simple linear regression, polynomial regression
- Parameter interpretation
- Least Square Method
- Gradient Descent Algorithm
- Coefficient of Determination, Mean Squared Error
- Decision boundary, classification rule
- Logistic regression, sigmoid function, probabilistic formulation
- Parameter interpretation
- Multi-class classification