Introduction to linear regression (Week 05 lecture notes)

Wei Q. Deng (Presented by Tianle Chen)

Department of Statistical Sciences

August 1st 2018

Topics covered in this lecture

- Machine learning the probabilistic way
- Supervised learning problems
- Regression
- Case studies in R and Python

Why and what is machine learning ?

- We are in the era of **big data**: data of massive size and complex nature
 - 40 billion indexed web pages
 - the entire genome of thousands of individuals
 - more than 160 million credit card transactions per hour
- Machine learning provides **automated methods** to process and analyze data
 - automatically detect pattern in data
 - use the pattern to predict future data
 - use the pattern to inform decision making
- · Many machine learning approaches rely on tools of probability theory

Types of machine learning

- Supervised learning (or predictive learning)
 - classification (output is categorical)
 - regression (output is continuous)
- Unsupervised learning (or descriptive learning)
 - clustering
 - dimension reduction
 - graph structure
 - matrix completion
- Reinforcement learning (learning to make decisions)

A general setting for supervised learning problems

Suppose we have some inputs X and some outputs y, the goal is to learn a mapping using a set of labelled set of input-output pairs $\{(X_i, y_i)\}_{i=1}^n$:

• also called a training dataset

To verify the quality of this mapping, we often use

• a testing data set containing (sometimes absent) just the inputs X_{test}

The established mapping provides

- a model that describes the relationship between the inputs and outputs
- a model that can be used to predict outputs in the testing data

Some examples of classification problems

When the outputs are categorical, these are known as the **classification** or **pattern recognition** problems:

- Handwritten digit recognition: postal offices process your letters automatically.
- How likely are you to get a new credit card approved?
- Identifying faces from images.

handwritten digit recognition

0000000000000000000 / \ \ \ / / / / / \ \ / / / / / / 2222222222222222 3**33**333333333333333333 **モフクコフ**フイ**クク**クフ**フア**クフフ в **999999999999** 9

handwritten digit recognition

- the data consist of
 - images of handwritten digits from 0 to 9
 - labels identify by a human
- the goal is to process images automatically without humans hand labelling
- the learning algorithm need to achieve "near-human" performance

credit card approval

- given attributes such as
 - age
 - past credit history
 - current financial status
 - number of credit cards owned
 - household income, etc.
- Credit card applications either accepted (+) or rejected (-)

credit card approval

- Data can be found here
- Some analysis in R here
- Some analysis in Python here

Some examples of regression problems

When the outputs are continuous or ordinal, these are known as the **regression** problems:

- How much can your neighbour's house go for in a market like this?
- Predict the trading value of the TSX index tomorrow or in a month given the current market condition and other available information.
- Predict the age of a current subscriber of the Globe and Mail based on their browsing history.
- Predict the amount of pollutant in a given postal code area with weather data, traffic condition and information on industrial activities in the area and nearby.

house price prediction



12/44

house price prediction

- the listing price of all other houses in your neigbourhood is public (realtor.ca)
- a number of factors influence the listing price in your neighbourhood
 - house types (detached, semi-detached, etc.)
 - living space (square feet)
 - number of bedrooms
 - number of bathrooms
 - distance to a good public school
 - mortgage policies
 - availability of inventory
 - overall desirability of the neighbourhood
 - other amendities nearby
- How much should your neighbour list their house for in this market?

Model TSX index

Market Summary > S&P/TSX Composite Index INDEXTSI: OSPTX

+ Follow

16,433.86 -21.87 (0.13%) +

Jul. 27, 12:03 p.m. EDT · Disclaimer



Model TSX index

- Index Characteristics such as constituents
- a number of economic factors influence the index
 - labor costs
 - interest rates
 - government policy
 - taxes
 - etc.
- recent news or economic developments
- Can you provide a one-day or one-month forecast for the TSX composite index?

How to approach these problems?

What does data look like?

For the simplest case, we look at a dataset with one input vector.

i	Х	Y	
1	3.2	6.95	
2	1.5	5.22	
3	2.4	6.46	
4	3.2	7.03	
5	5.4	9.71	
6	5.5	9.67	
7	6.1	12.69	
8	5.7	10.85	
9	2.8	5.21	
10	3.9	7.82	

For example, the third observation is $(X_3, Y_3) = (2.4, 6.46)$. For real data, usually you don't have the index *i* column as given in the table.

Visualize the data with a scatterplot



Observing the scatterplot

- Most data seem to fall near a line
- For bigger values of X, Y seems to be bigger as well
- In other words, X and Y seem to be positively related
- How can we find the line that "best" describes the relationship between X and Y?

A simple linear regression

- the output: an outcome variable Y
- one input: a predictor variable X
- We want to find a simple linear function of $X = (x_1, ..., x_n)$ that approximates $Y = (y_1, ..., y_n)$:
 - A simple linear function has the form f(X) = a + bX
 - "best" fit for a data point (i) is measured by a small distance $r_i = y_i f(x_i)$
 - How do we aggregate the residuals (r_i) to find the line of best fit?
- Least squares method

$$\operatorname{argmin}_{a,b} \sum_{i=1}^{n} r_i^2 = \operatorname{argmin}_{a,b} \sum_{i=1}^{n} (y_i - a - bx_i)^2$$

Solving the least squares problem

$$\frac{\partial \sum_{i=1}^{n} r_i^2}{\partial a} = 2 \sum_{i=1}^{n} r_i \frac{\partial r_i}{\partial a} = 0$$

and

$$\frac{\partial \sum_{i=1}^{n} r_i^2}{\partial b} = 2 \sum_{i=1}^{n} r_i \frac{\partial r_i}{\partial b} = 0$$

Plug in $r_i = y_i - f(x_i) = y_i - a - bx_i$ and solve the two equations simultaneously to obtain the least squares solutions.

$$\hat{a} = \bar{y} - \hat{b}\bar{x}$$

and

$$\hat{b} = \frac{\sum_{i=1}^{n} x_i y_i - \frac{1}{n} \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{\sum_{i=1}^{n} x_i^2 - \frac{1}{n} (\sum_{i=1}^{n} x_i)^2}$$

The output is approximated by a simple linear function has the form f(X) = a + bX

- *a* is the **intercept** (i.e. the value of the function when X = 0)
- *b* is the **slope** (the strength of the positive or negative relationship between input and output)

The regression coefficients obtained using the least squares method:

- â is the estimated intercept
- \hat{b} is the estimated slope

Directions of association

- If the linear relationship is absent, then \hat{b} should be fairly close to 0
- If $\hat{b} > 0$, a positive relationship is possible
- If $\hat{b} > 0$, a negative relationship is possible

Strength of association

- The strength of association between the input and output can be captured by **Pearson's correlation coefficient** *if* the input has only one variable (e.g. a simple linear regression model)
 - the correlation coefficient in a sample is denoted by $r \in [-1,1]$
 - the closer |r| is to 1, the stronger the relationship
 - between (-0.2, 0.2)
- The strength of association between the input and output can be captured by **Pearson's correlation coefficient** *if* the input has only one variable (e.g. a simple linear regression model)
 - usually we square it and use r^2
 - intepreted as the porportion of variance in Y that is explained by X

The objective of the remaining lecture is to solve real prediction problems using statistical programming. You should be able to:

- Setup the regression problem
- Produce initial visualization of data
- Obtain an estimated linear regression model in either R or Python
- Read the outputs from R or Python
- Make prediction or interpret the results

Case study of regression in R and Python

- Data "Cars93" from the MASS library (appeared in lecture 1)
- Use data to answer the question:
 - Does car with a bigger fule tank capacity necessarily have more horsepower?
 - For a car with fuel tank capacity of 27 US gallons, what is the maximum horsepower roughly?
 - What about a car with 9 US gallons?
 - What about a car with 16 gallons?

About the dataset

- Data from 93 Cars on Sale in the USA in 1993
- Input variable is "Fuel tank capacity" in US gallon
- Output variable is "Horsepower" (maximum horsepower)
- Randomly choose 10 cars/observations to be the testing data
- The remaining 83 cars will be used to build the linear model

Data visualization (R)



Data visualization (R)







Data visualization (Python)



Data visualization (Python)



Summarizing the data (R)

<pre>mean(Cars93[["Fuel.tank.capacity"]])</pre>	<pre>mean(Cars93[["Horsepower"]])</pre>
## [1] 16.66452	## [1] 143.828
<pre>median(Cars93[["Fuel.tank.capacity"]])</pre>	<pre>median(Cars93[["Horsepower"]])</pre>
## [1] 16.4	## [1] 140
<pre>sd(Cars93[["Fuel.tank.capacity"]])</pre>	<pre>sd(Cars93[["Horsepower"]])</pre>
## [1] 3.27937	## [1] 52.37441

round(cor(Cars93[["Fuel.tank.capacity"]],Cars93[["Horsepower"]]),3)

[1] 0.712

Summarizing the data (Python)

import pandas

/Library/Frameworks/Python.framework/Versions/3.5/lib/python3.5/importlib/_bootstrap.py:2
return f(*args, **kwds)

```
import pandas
Cars93 = pandas.read_csv("Cars93.csv")
                                              Cars93 = pandas.read csv("Cars93.csv")
import scipy
                                              import scipy
from scipy.stats.stats import pearsonr
                                              from scipy.stats.stats import pearsonr
                                              print(Cars93[["Horsepower"]].mean())
## /Library/Frameworks/Python.framework/Versions/3.5/lib/python3.5/importlib/_bootstrap.py:2
##
    return f(*args, **kwds)
                                              ## Horsepower
                                                                143.827957
                                              ## dtype: float64
print(Cars93[["Fuel.tank.capacity"]].mean())
                                              print(Cars93[["Horsepower"]].median())
## Fuel.tank.capacity
                         16.664516
## dtvpe: float64
                                              ## Horsepower
                                                                140.0
                                              ## dtype: float64
print(Cars93[["Fuel.tank.capacity"]].median())
                                              print(Cars93[["Horsepower"]].std())
## Fuel.tank.capacity
                         16.4
## dtvpe: float64
                                              ## Horsepower
                                                                52.37441
                                              ## dtype: float64
print(Cars93[["Fuel.tank.capacity"]].std())
                                                                                      33/44
```

Estimated linear model in R

summary(lm(Horsepower~Fuel.tank.capacity, data = Cars93))

```
##
## Call:
## lm(formula = Horsepower ~ Fuel.tank.capacity, data = Cars93)
##
## Residuals:
##
      Min
              10 Median 30
                                     Max
## -96.321 -21.915 -5.349 14.863 120.528
##
## Coefficients:
##
                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)
                    -45.613 19.968 -2.284
                                                 0.0247 *
## Fuel.tank.capacity 11.368 1.176 9.667 1.27e-15 ***
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 36.99 on 91 degrees of freedom
## Multiple R-squared: 0.5066, Adjusted R-squared: 0.5012
## F-statistic: 93.45 on 1 and 91 DF, p-value: 1.268e-15
```

Estimated linear model in Python

```
import pandas
import numpy as np
import stats
Cars93 = pandas.read_csv("Cars93.csv")
x = np.array(Cars93[["Fuel.tank.capacity"]])
y = np.array(Cars93[["Horsepower"]])
x.reshape((len(x), 1))
y.reshape((len(y), 1))
from sklearn.linear_model import LinearRegression
```

```
## /Library/Frameworks/Python.framework/Versions/3.5/lib/python3.5/impo
## return f(*args, **kwds)
## /Library/Frameworks/Python.framework/Versions/3.5/lib/python3.5/impo
## return f(*args, **kwds)
```

```
regression_model = LinearRegression()
regression_model.fit(x, y)
```

```
## /Library/Frameworks/Python.framework/Versions/3.5/lib/python3.5/site
## linalg.lstsq(X, y)
```

```
intercept = regression_model.intercept_[0]
```

```
35/44
```

Interpreting the results

- The model is written as "Horsepower~Fuel.tank.capacity"
- Regression Coefficients:

Predictors	Estimate	Std. Error	t value	Pr(>
(Intercept)	-45.613	19.968	-2.284	0.0247 *
Fuel.tank.capacity	11.368	1.176	9.667	1.27e-15 ***

• Significance codes:

Extremely Strong	Strong	Moderate	Weak	No evidence against H_o
0 ***	0.001 **	0.01 *	0.05 .	0.1-1

Interpreting the results - Cont'd

- Residual standard error: 36.99 on 91 degrees of freedom
- R-squared
 - Multiple R-squared: 0.5066
 - Adjusted R-squared: 0.5012
- F-statistic: 93.45 on 1 and 91 DF,
- p-value: 1.268e-15

What additional information do you observe?

Does car with a bigger fuel tank capacity (FTC) necessarily have more horsepower?

- There is extremely strong evidence of a car with larger FTC having more horsepower.
- For one US gallon increase in fuel tank capacity, the maximum horsepower increases by 11 on average.

Prediction problems:

- For a car with fuel tank capacity of 27 US gallons, what is the maximum horsepower roughly?
- What about a car with 9 US gallons?
- What about a car with 16 gallons?

Prediction in R

1 2 3 ## 261.32084 56.69841 136.27380

Prediction in Python

```
x_new = [[27], [9], [16]]
y_predict = regression_model.predict(x_new)
print(y_predict)
```

[[261.32083648] ## [56.69840591] ## [136.27379557]]

Interpret the results

- For a car with fuel tank capacity of 27 US gallons, the *predicted* maximum horsepower is roughly 261.
- For a car with fuel tank capacity of 9 US gallons, the *predicted* maximum horsepower is roughly 57
- For a car with fuel tank capacity of 16 US gallons, the *predicted* maximum horsepower is roughly 136

Are these results trustworthy?



Next class

- Linear regression as a statistical model
- Classic assumptions of linear regression
- More examples of linear regression
- Tests for equality of means as a special case of a simple linear regression
- Extending to multiple predictors in R