NYC DATA SCIENCE ACADEMY

Python Machine Learning

Multiple Linear Regression

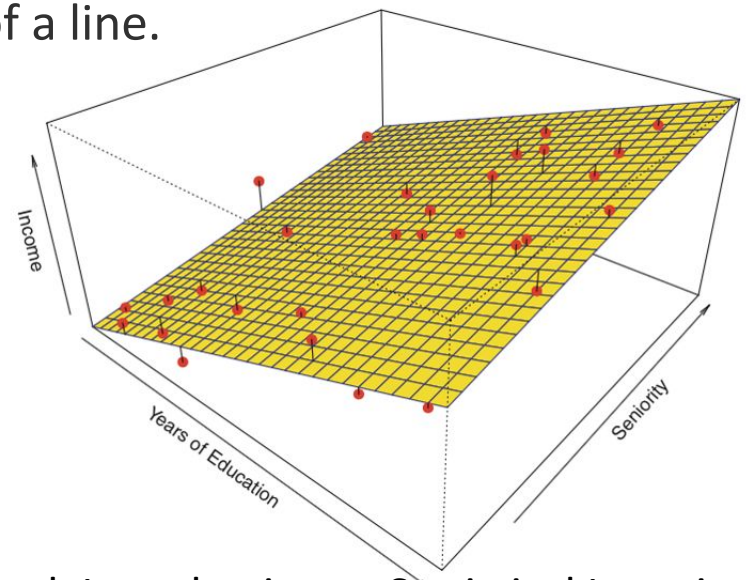
NYC Data Science Academy

Outline

- **Multiple Linear Regression**
  - Coefficient Estimation
  - Categorical Feature Dummification
  - Derived Feature Generation

Multiple Linear Regression

- In the simple linear regression lecture, we assume there is one input variable. In reality, the output *Y* usually depends on multiple input variables. Let's look at an example of two predictors.
- By looking at the data, it's reasonable to believe that *Income* depends on both *Years of Education* and *Seniority*. To visualize it, we need to plot a "plane" instead of a line.



Source: James et al. Introduction to Statistical Learning (Springer 2013)

What Is a Hyperplane?

- Geometrically, running a multiple linear regression corresponds to finding the 'optimal' hyperplane (2D plane in the income vs education & seniority example) approximating the scatter plot of the samples
- We will pay attention to the term **hyperplane**.
- A hyperplane is a codimension one linear geometric object in a high dimensional Euclidean space
- When the ambient space is two dimensional, a hyperplane is a line
- When the ambient space is three dimensional, a hyperplane is a 2D plane
- While we can visualize a line or a 2D plane, a high dimensional hyperplane is hard to be visualized

The Algebraic Expression of a Hyperplane

- Thus we will lean toward the algebraic expression of a hyperplane
- In algebraic terms, a hyperplane is described by a linear equation

$$\alpha_0 + \sum_i \alpha_i x_i = 0$$

- It is the geometric locus of all points which satisfy the above equation.
- In the context of linear regression, our ambient space is R^{p+1} with p being the number of input variables. Usually we isolate the last x variable and rename it y. The above linear equation can be re-casted into

$$y = \beta_0 + \sum_{i \leq p} \beta_i x_i$$

- β_0 is known as the intercept coefficient
- $\beta_i, 1 \leq i \leq p$ are known as the slope coefficients

Multiple Linear Regression

- If the output *Y* depends on more than one input variable, say X_1, X_2, … X_p, then we need to write the linear model as:

$$\hat{Y} = \hat{\beta}_0 + \sum_{i=1}^{p} \hat{\beta}_i X_i$$

- If we include a constant 1 in *X* and use the notation:

$$\beta = (\beta_0, \beta_1, \ldots, \beta_p)^T$$
$$X = (1, X_1, X_2, \ldots, X_p)$$

Then we can write the model in a simple matrix form:

$$\hat{Y} = X\hat{\beta}$$

Multiple Linear Regression

- To calculate the coefficients, again we use the least squares, but in a slightly different form.
- The RSS now can be written as:

$$RSS(\beta) = (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta)$$

  - A 1xn matrix multiplying an nx1 matrix produces a real number
- The task now is to minimize RSS(β). If X^TX is nonsingular, then we can prove that the minimum value of RSS(β) is obtained by:

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

- This is usually called the **normal equation** in the literature.
- The matrix multiplication operator $(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$ upon $\mathbf{y}$ is known as the **hat operator.**

Multiple Linear Regression

- Note: when two or more input variables are highly correlated with each other, then $X^T X$ is close to be singular and the solution will be unstable - any tiny fluctuations of data will cause huge changes in the model. The estimation errors of β will be large.
- When the feature matrix $\mathbf{X}$ is centered, (i.e. all the columns have mean value 0), the left hand side of the above normal equation is equal to

$$Cov(\mathbf{X}, \mathbf{X})^{-1} Cov(\mathbf{X}, \mathbf{y})$$

- When p is the number of features, $Cov(\mathbf{X}, \mathbf{X})^{-1}$ is the inverse pxp matrix of the feature covariance matrix.
- $Cov(\mathbf{X}, \mathbf{y})$ Is the p dimensional vector of feature-response covariance.

Multiple Linear Regression: Mathematically

- Task: find the estimates of $\beta_0, \beta_1, \beta_2, \ldots, \beta_p$ that reduce the sum of the squared vertical distances from the observations to the regression surface (i.e., the RSS) as much as possible.

- Procedure: derive formulas for these estimates using basic linear algebra.

Multiple Linear Regression: Mathematically

- Denote $\mathbf{X}$ as the n by $(p + 1)$ matrix with each row as an observation in our dataset; note that the first column will, by default, be a vector of 1's as a placeholder for the β_0 intercept term.
- Denote $\mathbf{y}$ as the n by 1 vector with each entry representing the response values in our dataset.

- We can rewrite the residual sum of squares in matrix notation as follows:

$$RSS(\beta) = (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta)$$

- This is a quadratic function with $(p + 1)$ parameters.

Multiple Linear Regression: Mathematically

- Differentiating with respect to β, we obtain the following:

$$RSS(\beta) = (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta)$$

$$\frac{\partial RSS}{\partial \beta} = -2\mathbf{X}^T(\mathbf{y} - \mathbf{X}\beta)$$

$$\frac{\partial^2 RSS}{\partial \beta \partial \beta^T} = 2\mathbf{X}^T\mathbf{X}$$

- Because the second derivative is necessarily positive, we know that this estimate is a minimum.

Multiple Linear Regression: Mathematically

- Assume that $\mathbf{X}$ is of full column rank, and thus that $\mathbf{X}^{\mathrm{T}}\mathbf{X}$ is positive definite. We can then set the first derivative equal to 0 as follows:

$$\frac{\partial RSS}{\partial \beta} = -2\mathbf{X}^T(\mathbf{y} - \mathbf{X}\beta) \stackrel{!}{=} 0$$

$$\Rightarrow \mathbf{X}^T(\mathbf{y} - \mathbf{X}\beta) = 0$$

$$\mathbf{X}^T\mathbf{y} - \mathbf{X}^T\mathbf{X}\beta = 0$$

$$\mathbf{X}^T\mathbf{y} = \mathbf{X}^T\mathbf{X}\beta$$

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

Multiple Linear Regression: Mathematically

- Thus, the least squares coefficient estimates for multiple linear regression are given by:

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

- Given our data, these are the best estimates for $\beta_0, \beta_1, \beta_2, \ldots, \beta_p$ as they ensure the sum of the squared vertical distances from the observations to the regression line (i.e., the RSS) is at a minimum.

- To obtain the fitted values of the regression, we simply pass our data matrix $\mathbf{X}$ through the matrix of coefficient estimates as follows:

$$\hat{\mathbf{y}} = \mathbf{X}\hat{\beta}$$

The Coefficient of Determination

- In simple linear regression lecture, we had introduced the concept of 'coefficient of determination'
- The same idea carries over to multiple linear regression
- When the linear model has vanishing slopes, it is essentially a constant which captures the average of the samples
- R square is defined to be 1 - RSS/TSS:
  - It is 0 when the intercept term is the mean value and the slopes are all zero
  - It is 1 when all the samples lie strictly within the hyperplane
  - In general it captures the ratio of explained variance

Outline

- **Multiple Linear Regression**
  - **Estimating Coefficients**
  - **The Issue of Multicollinearity**
  - Categorical Feature Dummification
  - Derived Feature Generation

Descriptive Statistics

- In addition to assuming the dependence of response *Y* on predictors *X* is approximately linear, linear regression makes several other key assumptions:
  - Statistical independence of errors
  - Homoscedasticity (constant variance) of the errors
  - **Low multicollinearity** in the predictors
- But in reality the data is very unlikely to satisfy all those assumptions.
- Failing to satisfy the assumptions may lead to a poor fit or even wrong results.

The Issue of MultiCollinearity

- We would like to study what's wrong if there are multiple features which are approximately multi-collinear?
- Suppose we run a multiple linear regression of a target w on a three-variate feature data x, y, z, finding the optimal regression equation to be $w \sim 2x - y + z$. We assume further that x, y and z satisfy an approximated linear relationship $x + y + z \cong 0$. Then it is sensible to use

  $2x - y + z + \alpha(x + y + z), \alpha \in \mathbf{R}$, to approximate w. In particular 3x+2z should be a good approximation of w.

- In other words, when some of the features are multicollinear, the linear model coefficients become ambiguous
- If the coefficients are less certain, it makes the model's prediction less trustworthy

MultiCollinearity -- Linear Algebra

- A more sophisticated angle to understand multicollinearity is through normal equation we have derived earlier
- It is known that when some collection of features are strictly multicollinear, the square matrix $X^T X$ in the **normal equation** become singular, which is NOT invertible
- When the **multicollinearity** is only approximately valid, the square matrix could be still invertible, but the model coefficients tend to be very unstable --- in the sense that the **confidence intervals** tend to blow up
- This results in estimating a poor model, with no resemblance of the true model that we are interested in

MultiCollinearity and Overfitting

- The goal of predictive task of machine learning involves training a model based on some known data and asks the trained model to predict the outcome on unseen data (not used in training)
- In the context of multi-linear regression, different collections of samples from the same underlying population can have different multi-collinearities: Those models which capture the training dataset multicollinearity can find high discrepancy between train-set performance vs test-set performance. This phenomenon is known as 'over-fitting'
- In the latter lecture, we will introduce the concept of **penalized linear regression** to tackle this problem
- Another important concept is called **feature selection.** By reducing the number of features, it reduces the possibility of multicollinearity

Boston House Data Set

- Boston house dataset is a small size house data of 506 samples with 14 columns, collected in 1970s Boston
- The target variable is the house price, while the input features are those which can potentially affect the prices
- The dataset is so popular that it is included in sklearn as a default data
- To load the data in scikit-learn, we call

```
from sklearn.datasets import load_boston

Boston = load_boston()
```

- Boston.data stores the numpy array of all features
- Boston.target stores the target, the house median prices

The Features of Boston House Dataset

Crim per capita crime rate by town.

Zn proportion of residential land zoned for lots over 25,000 sq.ft.

Indus proportion of non-retail business acres per town.

Chas Charles River dummy variable (= 1 if tract bounds river; 0 otherwise).

Nox nitrogen oxides concentration (parts per 10 million).

Rm average number of rooms per dwelling.

Age proportion of owner-occupied units built prior to 1940.

Dis weighted mean of distances to five Boston employment centres.

Rad index of accessibility to radial highways.

Tax full-value property-tax rate per $10,000.

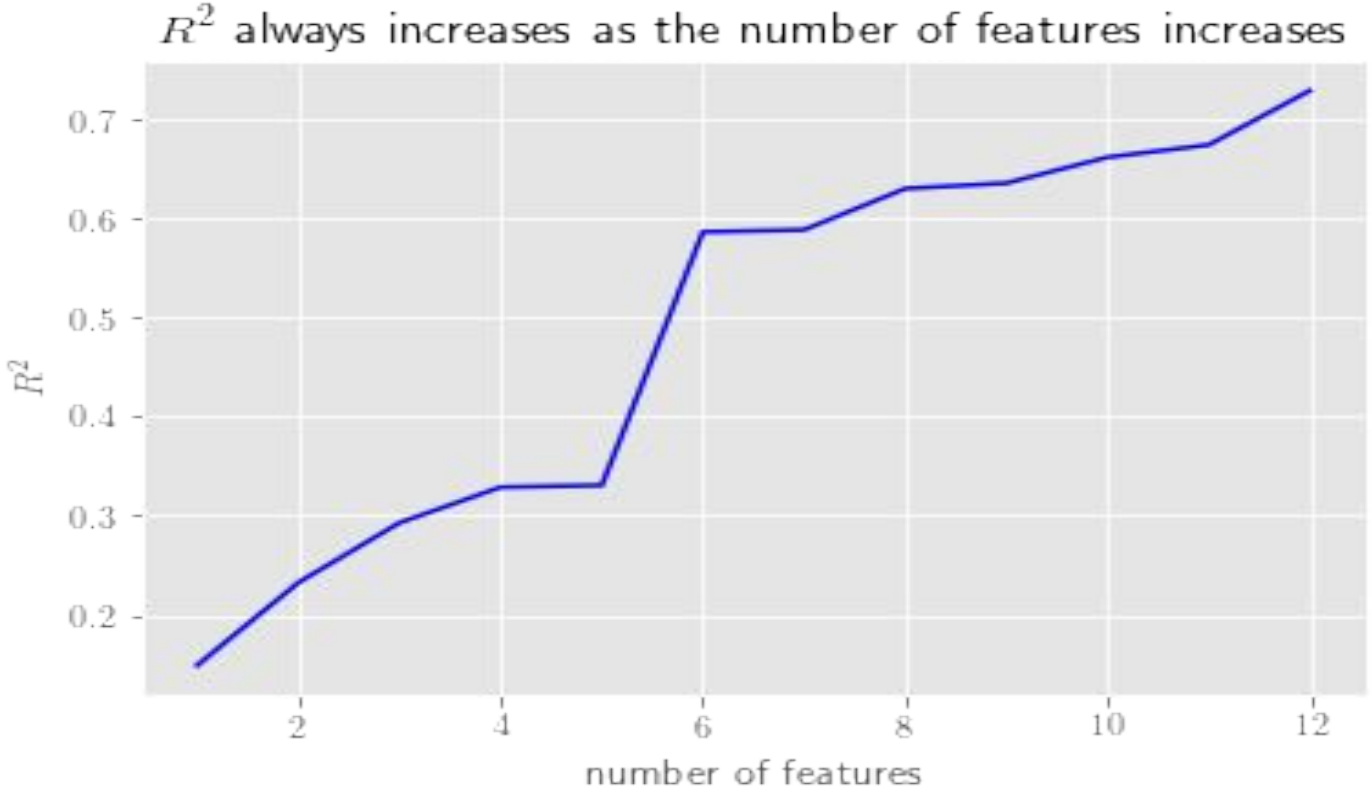
Ptratio pupil-teacher ratio by town.

Black 1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town.

Lstat lower status of the population (percent).

Boston House Data Set

- Increasing the number of features always increases the coefficients of determination



Why R Square is Not A Good Performance Measure Across Increasing Feature Counts

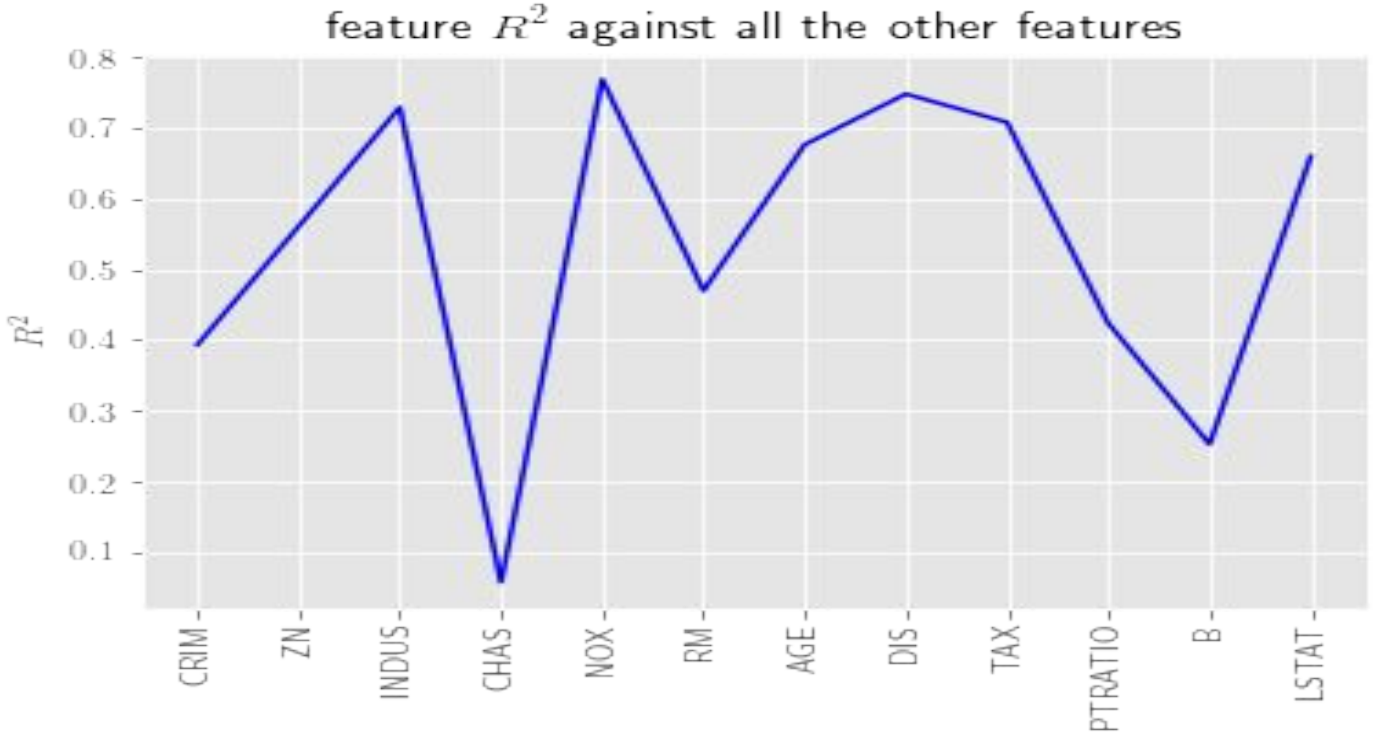
- By optimality, the multiple linear regression model has the least **RSS** among all the linear models with the same features and target
- Thus it has the highest **R square** compared to the other non-optimal linear models
- One of the non-optimal linear models consists of turning off one of the feature's slope--in effect reducing the number of features by 1
- This observation explains why adding more features always improves the **R square**

How To Detect MultiCollinearity?

- We know from the previous slides that multicollinearity among the features is detrimental to the multiple linear regression model
- But how do we detect whether, or how serious, is the multicollinearity?
- We can treat each feature as a new target and run a new **MLR** against all the other features (ignoring the original target variable)
- The **R square** of this new model should give us a good indication on the level of whether the chosen feature is linearly related to the others
- A higher **R square** score indicates a potential multicollinearity issue on this particular input feature
- Ideally when we scan through all the input features, low **R square** scores across the board is a good sign that the problem is well-controlled

Boston House Data Set

- Run a multiple linear regression of each feature against all the other features



- 1-R square (a feature vs the rest) is also known as tolerance
- A low tolerance indicates high multicollinearity!

Descriptive Statistics

- We will now explore some basic descriptive statistics to understand the variables before fitting the linear model.
  - Univariate analysis: understanding the distribution of a single variable (mean, median, quartiles, standard deviation, etc.)
  - Bivariate analysis: understanding the relationship between the pairs of variables (correlation, covariance, etc.)
- In the next few lectures, we will introduce some of the strategies that can transform poor data into one which fits a better model.

Outline

- **Multiple Linear Regression**
  - Estimating Coefficients
  - The issue of Multicollinearity
  - **Categorical Feature Dummification**
  - Derived Feature Generation

Categorical Input Variables

- Although we have seen categorical variables in our input, we have not used them in our regression models. However, they may be useful in making predictions:
  - For example, if we want to explore the relationship between smoking and lung cancer rate, the variable “smoking” usually has two categories, “yes” and “no”.
  - Categorical variables can be effectively coded as integers. For instance “yes”, “no” are often coded as 1 and 0.
- Categorical Values can be separated into two sub-types:
  - Ordinal categorical type
  - Nominal categorical type
- We will now discuss how to incorporate categorical variables in regression.

Categorical Input Variables

- If a categorical variable is binary, i.e., contains two categories, then we just need to code them as [0, 1]. In this scenario, 0 will be treated as the reference category.
- When there are more than two categories, often we cannot represent the variables by increasing the number of integers, as scikit-learn estimators expect continuous input, and would interpret the categories as being ordered (ordinal), which is often not desired.
- For example, if you convert the categorical variable [“Red”, “Blue”, “Green”] as [0, 1, 2], then scikit-learn linear regression will consider them as ordered numbers, instead of distinct categories. Such categorical features with no ordering relationship is called ‘nominal’.

Categorical Input Variables

- The most useful and commonly used coding is via dummy variables: a *K*-level categorical variable is represented by *K* binary variables, only one of which is on at a time. This is called *1-of-K encoding*.
- Scikit-learn provides a class called **OneHotEncoder** which encodes categorical variables using the 1-of-K scheme, but it only takes integers as input (i.e. the feature has to be pre-coded into integers).
- Dummifying categorical variables produces sparse data matrices, which can take up a large memory space. **OneHotEncoder** in the preprocessing submodule allows us to store it in a compressed dense format.
- In the next slide we will show you how to convert categorical variables using ***pandas*.get_dummies()**.

Categorical Input Variables

- Suppose a laptop sales data set has a feature called **web browser** which contains the types of pre-installed web browser for 5 different laptop models.

```
import pandas as pd
browser = pd.Series(["Safari","Chrome", "IE", "IE", "Safari"])
browser_dummy = pd.get_dummies(browser)
browser_dummy
```

	Chrome	IE	Safari
0	0	0	1
1	1	0	0
2	0	1	0
3	0	1	0
4	0	0	1

Categorical Input Variables

- In the previous example, we generated three dummy variables, as the number of categories is 3. For each single record we only have one “1” and the rest are all 0’s.
- We can consider one category as the base and drop it without losing any information since all the dummy variables are mutually exclusive.
- So the following dummy variables will be sufficient.

```
browser_dummy.drop('Chrome', 1)
```

	IE	Safari
0	0	1
1	0	0
2	1	0
3	1	0
4	0	1

Remark: Categorical Output Variables

- The idea of dummifying categorical input variables can be applied to classification task as well
- When the output is a categorical variable with multiple values, dummifying such a categorical variable can transform a multiclass classification task into multiple binary classification tasks
- We will discuss this when we discuss multi-class classification tasks

The Meaning of The Slopes of Dummified Features

- When we dummify a categorical feature into many binary columns and run a multiple linear regression against these dummified columns, it introduces a lot of linear regression slopes. What do these newly added slopes mean?
- For a continuous feature, the slope measures the increase of the target given a unit increase of the given feature, controlling all the other features fixed
- For a categorical feature which can take multiple values, the subsets of samples which take a particular value decompose the set of samples into a disjoint subsets
- The binary feature selects whether the sample is taking a particular value or not

The Meaning of Dummified Features Slopes

- The slope of the specified binary feature represents the jump on the intercept when we restrict the samples to this particular subset
- For example in a data set involving hypothetically the browser column, one runs a multiple linear regression involving the dummified **safari** and **IE** columns, and the slopes are 2, -1, respectively.
- Then the slopes 2 and -1 represent the increase and decrease of the intercept term of samples in **safari** and **IE** relative to the baseline **chrome** samples

Is Discrete Feature Dummification Always Necessary?

- In running a multiple linear regression, it is always necessary to dummify a **nominal** categorical feature
- Linear regression tries to fit a linear relationship between the feature and the continuous target. If we merely relabel the categorical feature as numbers (typically using integers), it will infer a non-existing relationship between the feature and the target
- On the other hand, it is not necessary to dummify an ordinal categorical feature. Instead one might need to transform the feature nonlinearly such that the transformed feature and the target satisfy a linear relationship
- Among advanced **ML** algorithms dummification is often not necessary for tree based models, which will be discussed in individual lectures

OutLine

- **Multiple Linear Regression**
  - Estimating Coefficients
  - The Issue of Multicollinearity
  - Categorical Feature Dummification
  - **Derived Feature Generation**

Generating Derived Features

- Up to know we focus on linear models, i.e. assuming that the target satisfies an approximately linear relationship with the input features
- When the linear assumption is not met, we may derive new nonlinear feature(s) based on the original set of features.
- For example, we may take different powers of a single feature, the product of two or multiple features, as our derived new feature
- For example, when running an **MLR** of the medical-bills on patients' heights/weights, it makes sense to generate a new feature called BMI = weight/(height*height). **BMI** has been used in health-risk assessment widely

Feature Engineering

- In training supervised machine learning models, we often need to introduce new features
  - Either by adding brand new features from outside sources
  - Or adding new features derived from the original features
  - Or using such new features to replace the original features
  - Eliminate any unnecessary features
- Such action is aggregately called **Feature Engineering**
- Successful **feature engineering** involves high creativity

The Danger of Using Too Many Features

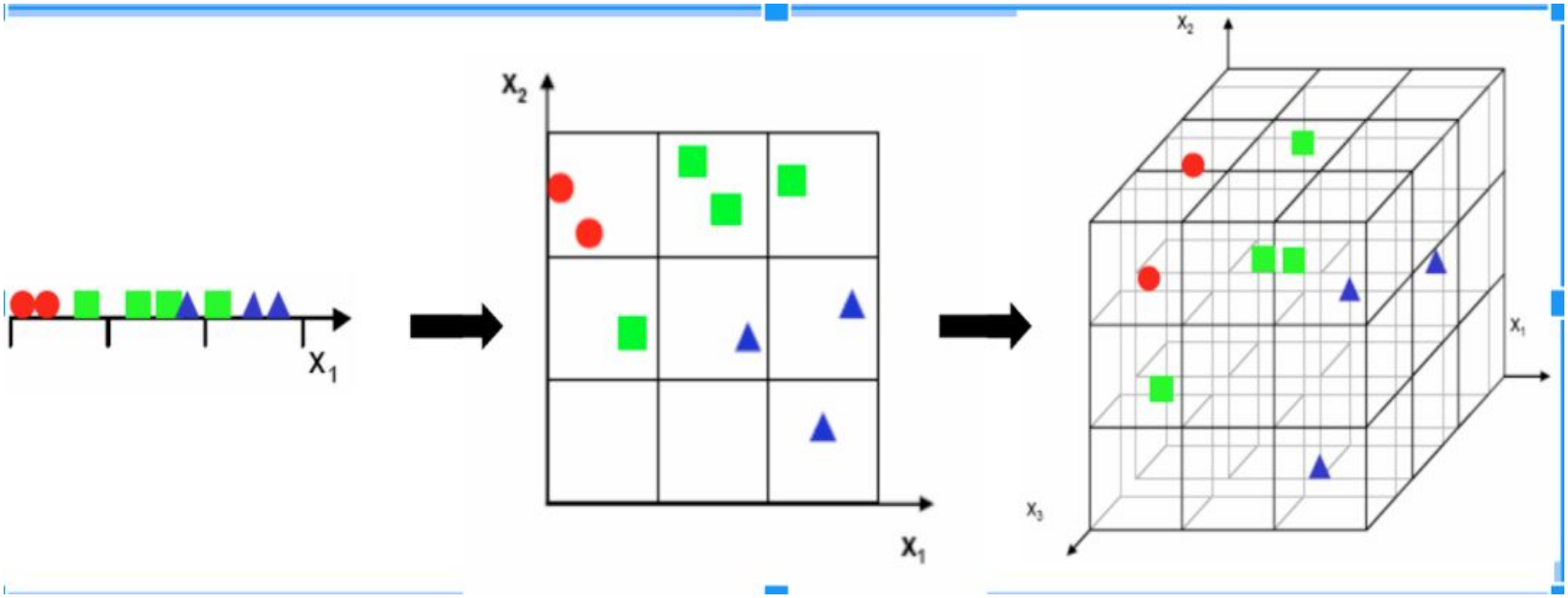
- On the flip side, it is dangerous to generate too many new features into a very high dimensional multi-linear regression
- On the one hand, adding more and more features always increases the coefficient of determination (so it looks good superficially)
- On the other hand, when more and more features are added it is more likely for some of these features to satisfy an approximated linear relationship -- This is a lightning rod for creating a poor model
- From the point of view of **normal equation**, this involves estimating the high dimensional covariance matrix Cov(X,X) with little data, which is very likely to fail

The Curse of Dimensionality

- In machine learning/statistical learning, there is a well known phenomenon that most of the machine learning models perform poorly when the feature dimension grows very large
- It is known as '**the curse of dimensionality**'
- The **curse of dimensionality** is due to the sparsity of the data in high dimension, it becomes harder and harder for a statistical model to be estimated properly by a finite set of data points of fixed cardinality
- Put it differently, the number of data points which are needed to estimate a statistical model in high dimensional feature space grows exponentially with respect to the dimension

The Sparsity of the Data

- The number of identical cubes of size 1/N in a D dimensional unit cube is equal to N^D. Apparently it blows up when the dimension goes large
- As different data points can spread out in different sub cubes, it is less likely for them to be neighbors to each other--hence the sparsity



Multiple Linear Regression and the Curse of Dimensionality

- When the number of feature increases, the coefficient of determination always increases
- It is because the new features can be always used to explain the pattern in the residuals what the original feature set is not capable of explaining
- On the other hand, the increasing of the dimensionality increases the chance of multicollinearity, which leads to an inflated confidence interval/model variance on the regression coefficients
- As a side effect, the true model could be very far away from the estimated model. Or in other words, the coefficients of the estimated model provides very a poor approximation to the reality
- This is linked to curse of dimensionality by the normal equation

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

The Difficulty of Estimating the Samples' Covariance Matrix

- When the features are properly demeaned, the normal equation is a finite sample approximation of the following quotient

$$Cov(\mathbf{X}, \mathbf{X})^{-1} Cov(\mathbf{X}, \mathbf{y})$$

- When the feature dimension is p, the covariance matrix Cov(X,X) is a square symmetric pxp matrix. Such matrices have p(p+1)/2 free parameters to estimate
- Given a fixed number of samples, it becomes harder and harder to estimate so many parameters: For example, for p = 1000, the finite number of data samples is used to estimate about ~ 500000, half a million, covariance matrix coefficients

The Relationship Between Simple/Multiple Linear Regressions

- Is running a multiple linear regression equivalent to running multiple simple linear regressions against each of these features?
- In general it is not the case
- They become equivalent only when the features are mutually uncorrelated to each other
- When all the features are mutually uncorrelated to each other, the covariance matrix Cov(X,X) becomes diagonal, and the MLE normal equation reduces to multiple simple linear regression normal equation
- In the real world application, it is rare that different features are uncorrelated to each other

Confidence Intervals of MLR Coefficients

- The off-diagonal elements of the covariance matrix Cov(X,X) measure the level of correlations between each pair of features
- When a feature is multicollinear with some other list of features, it also produces some nonvanishing off-diagonal terms of the covariance matrix
- Thus multicollinearity among features affect the estimation of the slope coefficients
- Just like simple linear regression, there are regression coefficients' standard errors and confidence intervals
- The ranges of the confidence intervals decide the range where the true model's coefficients most likely to lie within
- High multicollinearity widens the confidence intervals and reduces the effectiveness of the linear model estimation

Hands-on Session

- Please go to the **"Multiple Linear Regression in Scikit-Learn"** in the lecture code.