DSO530 Statistical Learning Methods

Lecture 2b: Multiple Linear Regression

Dr. Xin Tong
Department of Data Sciences and Operations
Marshall School of Business
University of Southern California
xint@marshall.usc.edu

Multiple linear regression

In addition to scalar input $x \in \mathbb{R}$, we can consider vector input $x \in \mathbb{R}^p$

- p is feature dimension of input (sometimes use d for *dimension*)
- Inputs of form

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{bmatrix}$$

multiple predictors used.

- Call x the *covariates*, *independent variables*, *explanatory variables*, *features*, *attributes* or *predictor variables*
  - Note that variables are usually NOT independent of one another

Example: housing data

- Output (response, target, dependent) variable is medv, the median home price in different neighborhoods
- covariates include
  - crim: per capita crime rate
  - rm: average number of rooms per dwelling
  - zn: proportion of large lots (zoned for $> 25,000$ feet)
  - river: whether a home is near a river ($x \in \{0, 1\}$)
  - ptratio: pupil-teacher ratio by town
- Let's do some data exploration

```
housing = pd.read_csv("data/housing.csv"); display(housing.head())
```

	crim	zn	river	rm	ptratio	medv
0	0.00632	18.0	0	6.575	15.3	24.0
1	0.02731	0.0	0	6.421	17.8	21.6
2	0.02729	0.0	0	7.185	17.8	34.7
3	0.03237	0.0	0	6.998	18.7	33.4
4	0.06905	0.0	0	7.147	18.7	36.2

1,000s $ $

Figure 1: Import data

Check data

```
housing.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 6 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   crim     506 non-null    float64
 1   zn       506 non-null    float64
 2   river    506 non-null    int64  
 3   rm       506 non-null    float64
 4   ptratio  506 non-null    float64
 5   medv     506 non-null    float64
dtypes: float64(5), int64(1)
memory usage: 23.8 KB
```

```
import seaborn as sns
sns.histplot(housing['medv'])
plt.show()
```

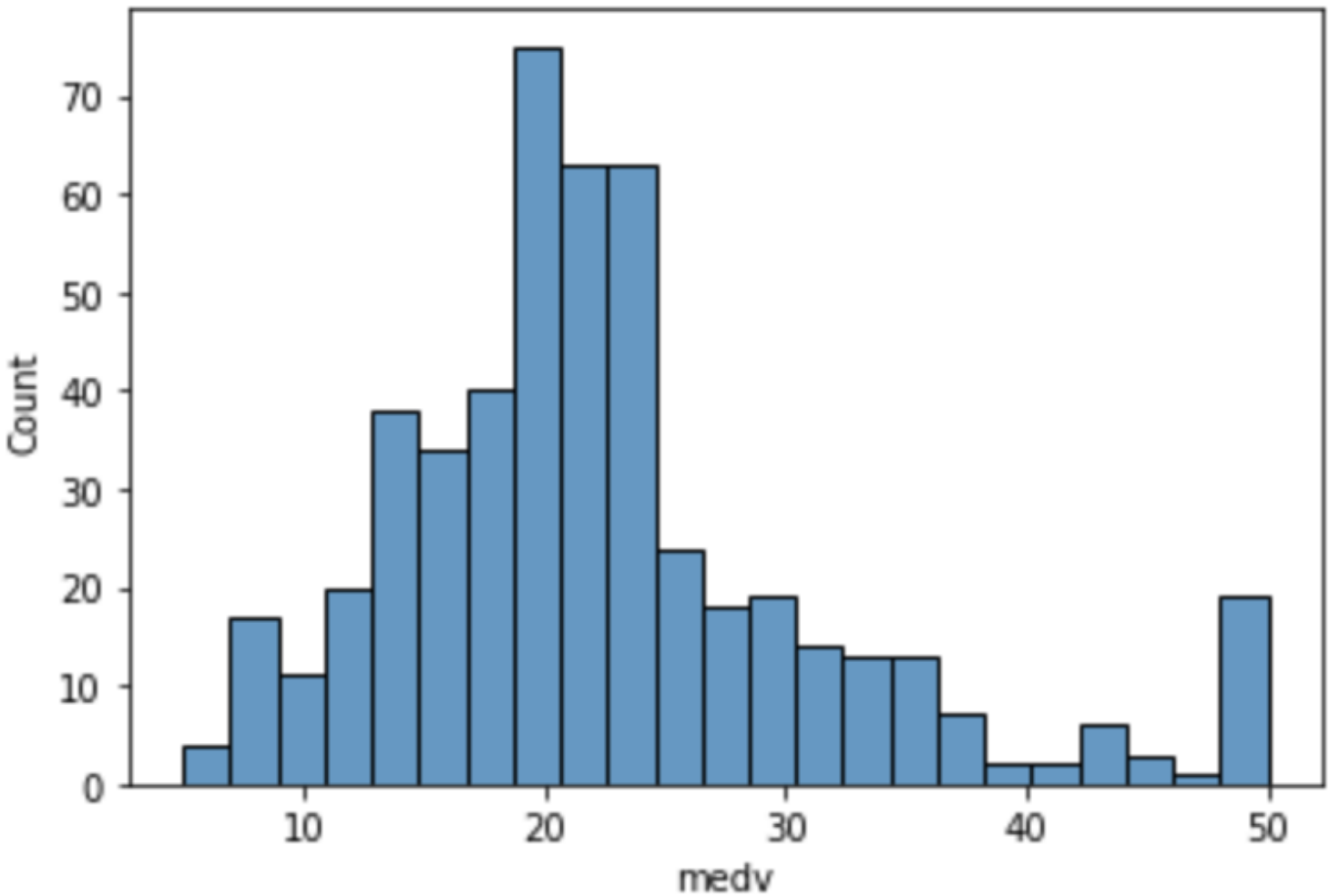


Figure 2: Plot response variable

```
correlation_matrix = housing.corr().round(2)
print(correlation_matrix)

##or alternatively do the following
## sns.heatmap(data=correlation_matrix, annot=False)
# where "annot = True" to print the values inside the square
```

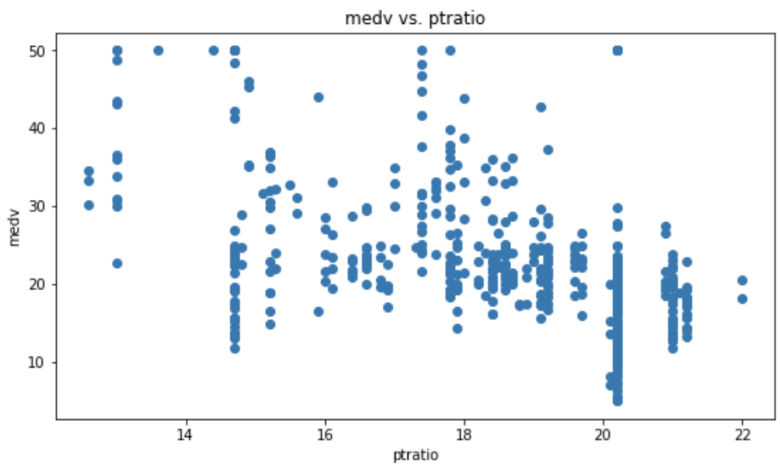
	crim	zn	river	rm	ptratio	medv
crim	1.00	-0.20	-0.06	-0.22	0.29	-0.39
zn	-0.20	1.00	-0.04	0.31	-0.39	0.36
river	-0.06	-0.04	1.00	0.09	-0.12	0.18
rm	-0.22	0.31	0.09	1.00	-0.36	0.70
ptratio	0.29	-0.39	-0.12	-0.36	1.00	-0.51
medv	-0.39	0.36	0.18	0.70	-0.51	1.00

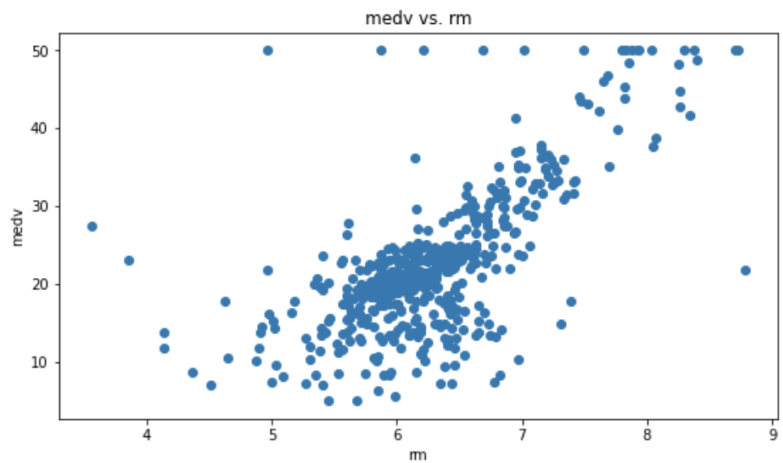
Figure 3: Correlation matrix

characteristics for building best model with 2 features.

- For illustration purpose we only use `ptratio` and `rm` as covariates
- It looks like one has negative correlation with `medv` and the other has positive correlation with `medv`
- If for some reason, we have to choose only one of these two predictors, which one would you like to choose?

```
plt.figure(figsize=(20, 5))
features = ['ptratio', 'rm']; target = housing['medv']
for i, col in enumerate(features):
    plt.subplot(1, len(features) , i+1) # subplot(nrows, ncols, index). Three separate integers describing
    #the position of the subplot. If the three integers are nrows, ncols, and index in order,
    #the subplot will take the index position on a grid with nrows rows and ncols columns.
    #index starts at 1 in the upper left corner and increases to the right.
    x = housing[col]; y = target
    plt.scatter(x, y)
    plt.title('medv' + " vs. " + col)
    plt.xlabel(col); plt.ylabel('medv')
```





```
X2 = housing[['ptratio','rm']].values
y2 = housing['medv'].values
linear_model_2 = LinearRegression(); linear_model_2.fit(X2, y2)
r_sq_house = linear_model_2.score(X2, y2); print('coefficient of determination:', r_sq_house)
```

↳ In sample R^2, replace X_2, y_2 for out of sample R^2

```
coefficient of determination: 0.5612534621272915
```

```
print(linear_model_2.coef_); print(linear_model_2.intercept_)
```

```
[-1.2671614   7.71407021]
-2.5611648168335925
```

Figure 4: Fitting a multiple linear regression model

- The model fitting is not much different from fitting a simple linear regression model
- Except that one does not need to worry about the covariates being a 1D object
- Try regress `medv` on `rm` only, do you see a larger or smaller R square?
- The Python `sklearn` library focuses on prediction, is not yet fully developed for some typical statistical inferential tasks.

↳ because larger model includes smaller model w/ just 1 predictor.

Scenario:
① $Y \sim X_1$
② $Y \sim X_2, X_3$
} predictors are completely different therefore depends on the predictors, does not depend on whether predictors are significant or not

```
import statsmodels.api as sm
X3 = sm.add_constant(X2)
ols = sm.OLS(y2, X3)
ols_result = ols.fit()
ols_result.summary()
```

{ Statsmodels allows us to view coefficients more easily }
w/o this line, we force $\beta_0 = 0$

OLS Regression Results

OLS : Ordinary Least Squares.

Dep. Variable:	y	**R-squared:**	0.561
Model:	OLS	**Adj. R-squared:**	0.560
Method:	Least Squares	**F-statistic:**	321.7
Date:	Sun, 31 Jan 2021	**Prob (F-statistic):**	1.04e-90
Time:	20:23:15	**Log-Likelihood:**	-1631.8
No. Observations:	506	**AIC:**	3270.
Df Residuals:	503	**BIC:**	3282.
Df Model:	2		
Covariance Type:	nonrobust		

⇒ adjusts for the model size.

3 Imp metrics:

① In Sample R^2
Used to compare models of w/ same no. of predictors.

② Out of Sample R^2
Used to compare models of different size.

③ Adj. R^2
Used to compare models of different sizes (same size also O.K.) but different no. of predictors.

- continue from the previous page. The summary also contains:

	coef	std err	t	P>|t|	[0.025	0.975]
const ($\hat{\beta}_0$)	-2.5612	4.189	-0.611	0.541	-10.791	5.669
x1 ($\hat{\beta}_1$)	-1.2672	0.134	-9.440	0.000	-1.531	-1.003
x2 ($\hat{\beta}_2$)	7.7141	0.414	18.650	0.000	6.901	8.527

for β_0 not $\hat{\beta}_0$

- Compare the coefficients fitted by `statmodels` and `sklearn`
- What are the second to the last columns about?
- We see some t-statistics and their p-values here. What are the null hypotheses?

In Sample & Adj give same rankings for models of same size
Adj R^2: valid only for linear models.

- continue from the previous page. The summary also contains:

	coef	std err	t	P>|t|	[0.025	0.975]
const	-2.5612	4.189	-0.611	0.541	-10.791	5.669
x1	-1.2672	0.134	-9.440	0.000	-1.531	-1.003
x2	7.7141	0.414	18.650	0.000	6.901	8.527

This confidence interval is for β_0, not $\hat{\beta}_0$ (parameter not point estimate)

$t = \frac{\text{coeff}}{\text{std error}}$

$H_0: \beta_i = 0 \quad H_a: \beta_i \neq 0$

if p-val this small, α does not matter

↳ Level of significance.

- Compare the coefficients fitted by `statmodels` and `sklearn`
- What are the second to the last columns about?
- We see some t-statistics and their p-values here. What are the null hypotheses?

When we use more than one input variables

- Make predictions

$$\hat{y} = \hat{\beta}_0 + \sum_{j=1}^{p} \hat{\beta}_j x_j$$

- Model the data as

$$y = \beta_0 + \sum_{j=1}^{p} \beta_j x_j + \varepsilon$$

# of parameters = $p+2$

$\varepsilon \sim N(0, \sigma^2)$

$\hookrightarrow (p+2)^{th}$ model.

when σ is smaller, pts will be closer to hyper plane.

where ε is noise

- As in simple linear regression, choose noise distribution based on
  - Convenience (assume it is normally distributed)
  - Prior knowledge (rarely)

Fitting a multiple regression

- Making predictions using

$$\hat{y} = \hat{\beta}_0 + \sum_{j=1}^{p} \hat{\beta}_j x_j$$

- Fit as in single variable case. Solve (least squares criterion or OLS)

$$\underset{\beta_0 \in \mathbb{R}, \beta \in \mathbb{R}^p}{\text{minimize}} \sum_{i=1}^{n} (y_i - \beta_0 - \beta^T x_i)^2 = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

- Since each observation has p values, $x_i = (x_{i1}, \cdots, x_{ip})^T$, where $i = 1, \cdots, n$. In matrix notation, let $y = (y_1 \cdots, y_n)^T$ and

$$X = \begin{bmatrix} 1 & x_{11} & x_{12} & x_{13} & \cdots & x_{1p} \\ 1 & x_{21} & x_{22} & x_{23} & \cdots & x_{2p} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & x_{n3} & \cdots & x_{np} \end{bmatrix}$$

where x_{ij} is the ith observation of the jth variable

Is there a relationship between the reponse and predictors?

↳ Same as asking, are some predictors good in predicting the response.

- $H_0 : \beta_1 = \cdots = \beta_p = 0$
- H_a: at least one of β_j is non-zero (useful)
- Did we miss β_0? ⇒ No, if everything is 0, b_0 would be the average.
- This hypothesis test is performed by computing the F-statistic,

If RSS ↓, f stat good up.

better fitting of data

$$F = \frac{(TSS - RSS)/p}{RSS/(n-p-1)}$$

num ↑ denom ↓

- The larger the F-statistic, the stronger evidence against null hypothesis
- Why do we need F-statistic and its p-values? Because inspecting individual t-statistic and p-value is NOT enough to answer the question posted on the top of the slide
- F-statistic is used only when p is small

$Y \sim X_1$
$Y \sim X_2$
$Y \not\sim X_1, X_2$ } this happens due to multicolinearity → In this case t is not sig but F is sig, therefore indication to drop one predictor.

RSE and R^2 for multiple linear regression

- RSE and R^2 are common measures of model fit.
- R^2 is used most often.
- R^2 will always increase when more variables are added to the model
- Reason: adding another variable to the least squares equations allows us to fit the training data more accurately
- Residual standard error (RSE)

$$RSE = \sqrt{\frac{1}{n-p-1}RSS}$$

- Does RSE always increase/decrease when more variables are added to the model?
- RSE has the same unit as the response. So we cannot compare RSEs across different responses.
- $RSE/\bar{y}$ could be more interpretable than just RSE.
- In this course, we focus on R^2

Making prediction

adj $R^2 = 1 - (1-R^2) \cdot (n-1)/(n-p-1)$

① $p = 1$ adj $R^2 \neq R^2$

② can adj $R^2 < 0$? Yes, if $1-R^2 > \frac{n-p-1}{n-1}$

Assumption: $\alpha R^2 < 1$

- There are three sorts of uncertainty associated with making prediction using linear model
  - the least squares plane

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1, + \cdots + \hat{\beta}_p x_p$$

    is an estimate for the true population regression plane

$$f(x) = \beta_0 + \beta_1 x_1, + \cdots + \beta_p x_p$$

  - assuming a linear model for $f(X)$ is almost always an *approximation* of reality
  - Even if you know $f(x)$, the response value cannot be determined perfectly because of the random error ε in the model ($y = f(x) + \varepsilon$)
  - The first two kind of errors are *reducible errors* and the last is *irreducible error*
- Why is (the same level) prediction interval (PI) for y wider than confidence interval (CI) for $f(x)$?

PI vs. CI

PI: prediction interval $\quad \hat{y} = \hat{\beta}_0 + \hat{\beta}_1 \text{ptratio} + \hat{\beta}_2 \cdot \text{rm}$

$[y_1, y_2]$ $\xrightarrow{\text{for a particular pt.}}$ ptratio $= 20$, rm $= 7$

CI: confidence interval for the mean response

CI is narrower than PI

$x_1 \dots x_n \sim N(\mu, \sigma^2)$

$\text{Var}(X_i) = \sigma^2$

$\text{Var}(\bar{X}) = \frac{\sigma^2}{n}$ } therefore variance reduces as sample size $\uparrow$.

Categorical (qualitative) predictors and interaction terms

n-1 dummy variables.

Categorical predictors:

- Sometimes, categorical variables can be useful in making predictions.
- We usually code qualitative variables by *dummy variables* (variables taking values 0 and 1). Why are they called dummy variables?
- Suppose you have a categorical variable with three *levels*. We need to code it with two dummy variables. See Python Tutorial 2 for more details

Interaction terms:

- Sometimes, influence of the predictors can not be decoupled into additive terms. In addition to *main effects*, we need *interaction effect*:

$$sales = \beta_0 + \beta_1 \cdot TV + \beta_2 \cdot radio + \beta_3 \cdot (radio \cdot TV) + \varepsilon$$

-Rewrite the above equation as:

$$sales = \beta_0 + (\beta_1 + \beta_3 \cdot radio) \cdot TV + \beta_2 \cdot radio + \varepsilon$$

- How do we interpret β_3? the increase in the effectiveness of `TV` advertising for a one unit increase in `radio` advertising (or vice-versa)

Nonlinear relationship

- True relationship between response and predictors might be nonlinear
- A first extension to linear regression to handle non-linear relationship is to use *polynomial regression*
- When we use higher order polynomial, **overfitting** is a concern.
- Suppose we use a polynimal of degree 5 to fit a data set with 6 points.
- All 6 residuals are zero. R^2 is 1. But this is clearly not the model you want.

Collinearity

- Collinearity: two or more predictor variables are closely related to one another
- Collinearity causes problems to separate out the individual effects of collinear variables on the response
- multicollinearity: collinearity that exists between three or more variables; hard to detect by looking at the correlation matrix
- variance inflation factor(VIF):

$$VIF(\hat{\beta}_j) = \frac{1}{1 - R^2_{X_j|X_{-j}}}$$

X_j as response & x_{j-1} & x_{j+1} as predictors.

where $R^2_{X_j|X_{-j}}$ is the R^2 from a regression of X_j onto all other predictors

- A VIF value larger than 5 or 10 indicates problematic amount of collinearity. Solution 1: drop variable with high VIF; solution 2: eliminate predictor with the highest p-value in its t-test
- Does collinearity matter for prediction? Not as much as for inference

Yes, but ↗ interpretation of coefficients

Performance measure on the (training) dataset does not tell the whole story

- So far, all the performance evaluations are on training data
- They tell us how well the model performs on the SAME data that was used to fit it
- By overfitting the model (e.g., using lots of predictors), we can make the RSS arbitrarily low.
- But this model will not accurately predict future datasets. Prediction of future datasets is called "generalization".
- For fixed amount of data, the larger the model, the larger the R^2. Therefore, R^2 cannot be used to compare models with different sizes.
- There are two solutions
  - create model-size adjusted performance measures, such as adj-R^2.
  - evaluate the performance on data NOT used for fitting (training).

Some quotes from a practitioner regarding linear regression

- Must clean data beforehand. Raw data often contains missing values, zeros values and erroneous extreme values. Don't let these values distort regression results, make sure no "garbage-in-garbage-out" scenario.
- Linear regression, like all other methodologies, is just a tool. What matters more is how to find more signals (x) that makes a stronger relationship. Nowadays there are many, many advanced modeling techniques, which can potentially improve the rsq over linear regression. But my advice is to first make enough effort in looking for new significant signals, and then explore newer methodologies.
- Ratio of number of samples to number of parameters: as a rule of thumb it's 30 to 1 but in practice the comfort level does depend on whether the signals are strong enough. For significant relationships 30 to 1 should be enough, for noisy data like financial data, the rsq is usually smaller than 10% or sometimes even smaller than 3%. In this case one really need more samples, like 100 to 1, to be comfortable not to get into spurious relationships.