510 DATA SCIENCE

Lecture 09

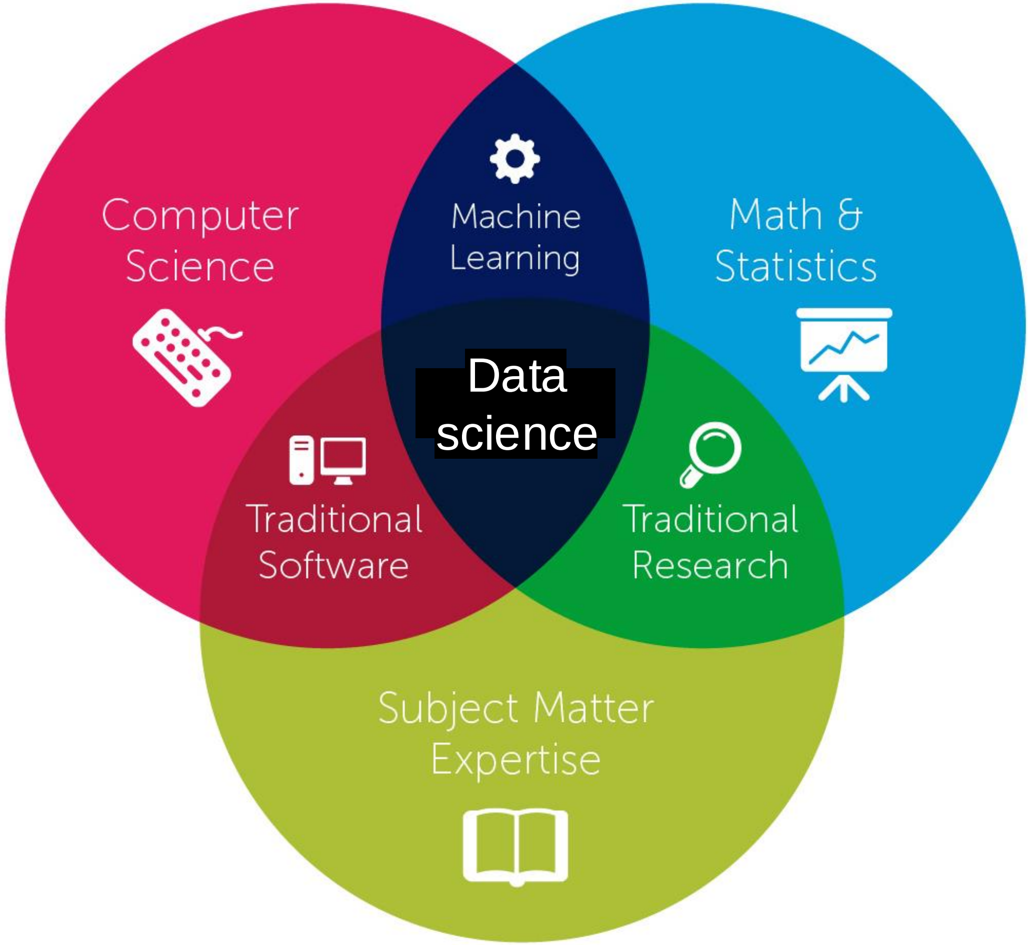
Fall 2024

Instructor: Assoc. Prof. Şener Özönder

Email: sener.ozonder@bogazici.edu.tr

Institute for Data Science & Artificial Intelligence

Boğaziçi University

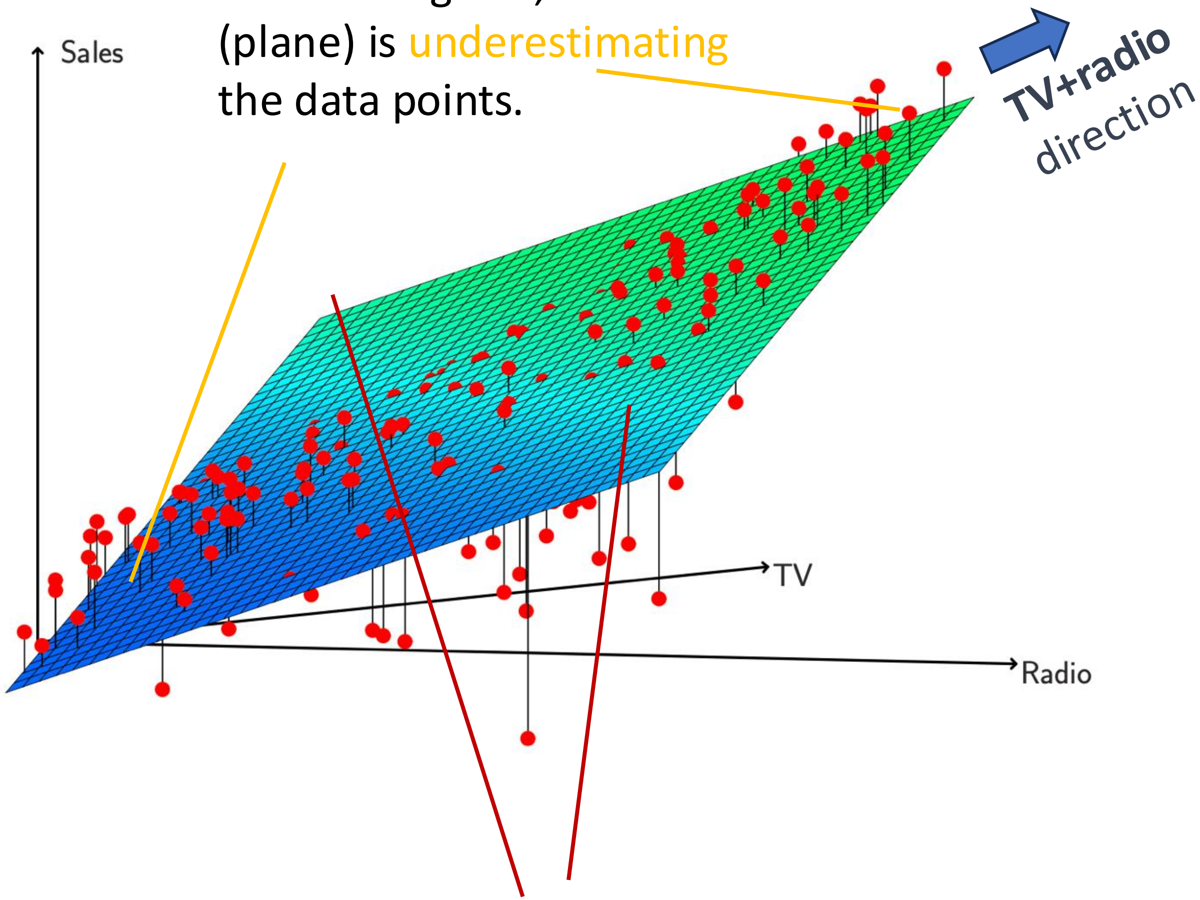


Linear Regression: Interaction terms

- Here's the plot of the best model we found so far **sales** = β_0 + β_1 **TV** + β_2 **radio**
- But the residues (distance between data points and the model plane) have a *non-planar* trend in the **TV+radio** direction. We can capture this behavior of data by adding an interaction term to model

$$\textbf{sales} = \beta_0 + \beta_1\,\textbf{TV} + \beta_2\,\textbf{radio} + \boxed{\beta_3\,\textbf{TV} \times \textbf{radio}}$$

- Let's first consider how such terms change the plane. The model will be no longer a perfect plane but a place with a curved surface at the corners.



At these regions, the model (plane) is underestimating the data points.

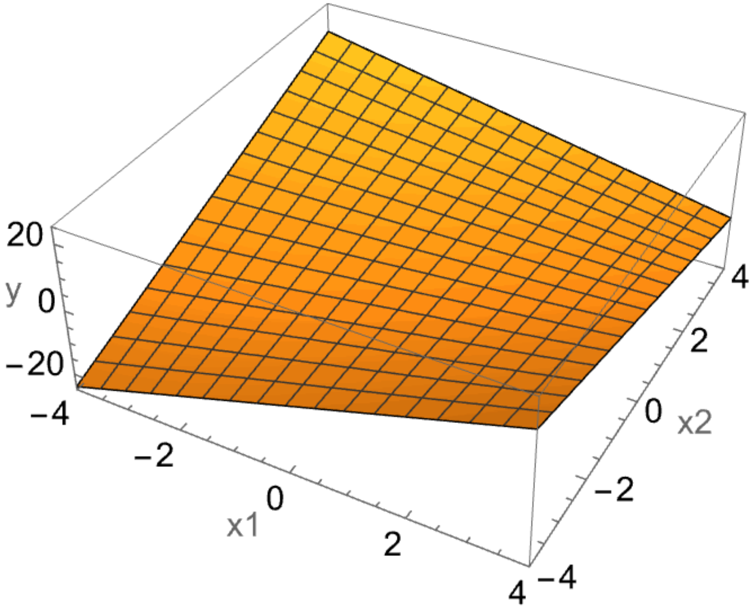
At these regions, the model (plane) is overestimating the data points.

Linear Regression: Interaction terms

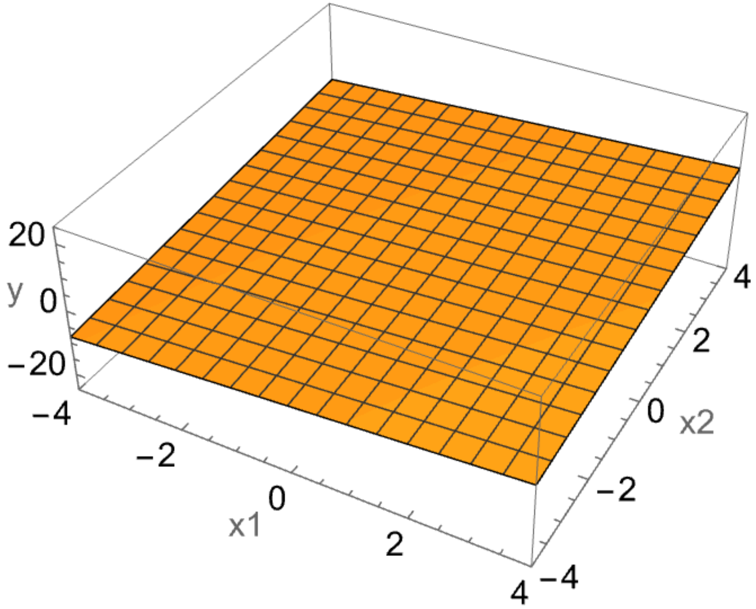
Interaction terms: $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 \boxed{x_1 x_2} + \ldots + \varepsilon_i$

- $x_1 x_2$ is called interaction term because two predictors are multiplied. It has nothing to do with correlation between predictors x_1 and x_2 .
- The interaction term turns the plane into a curved surface.

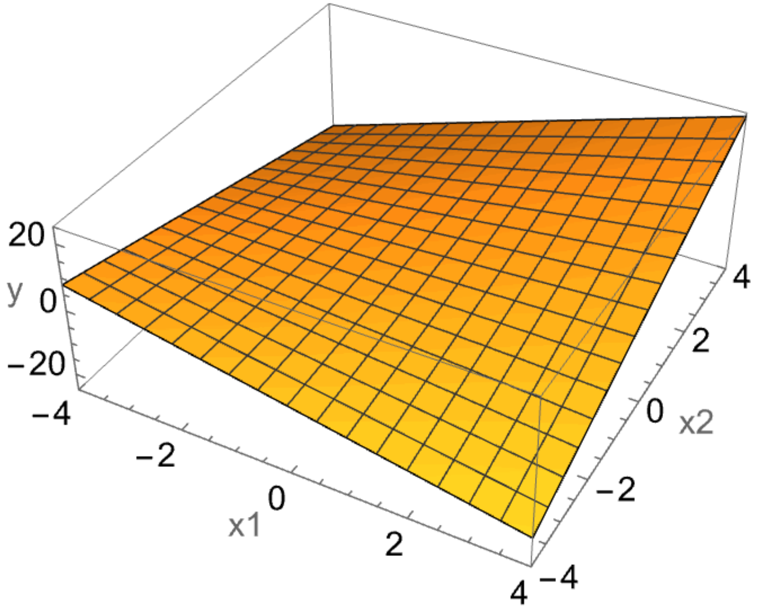
$\beta_3 = -1$



$\beta_3 = 0$



$\beta_3 = +1$

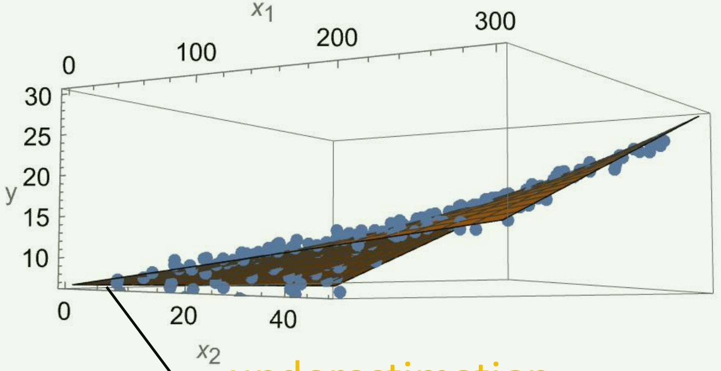


Linear Regression: Interaction terms

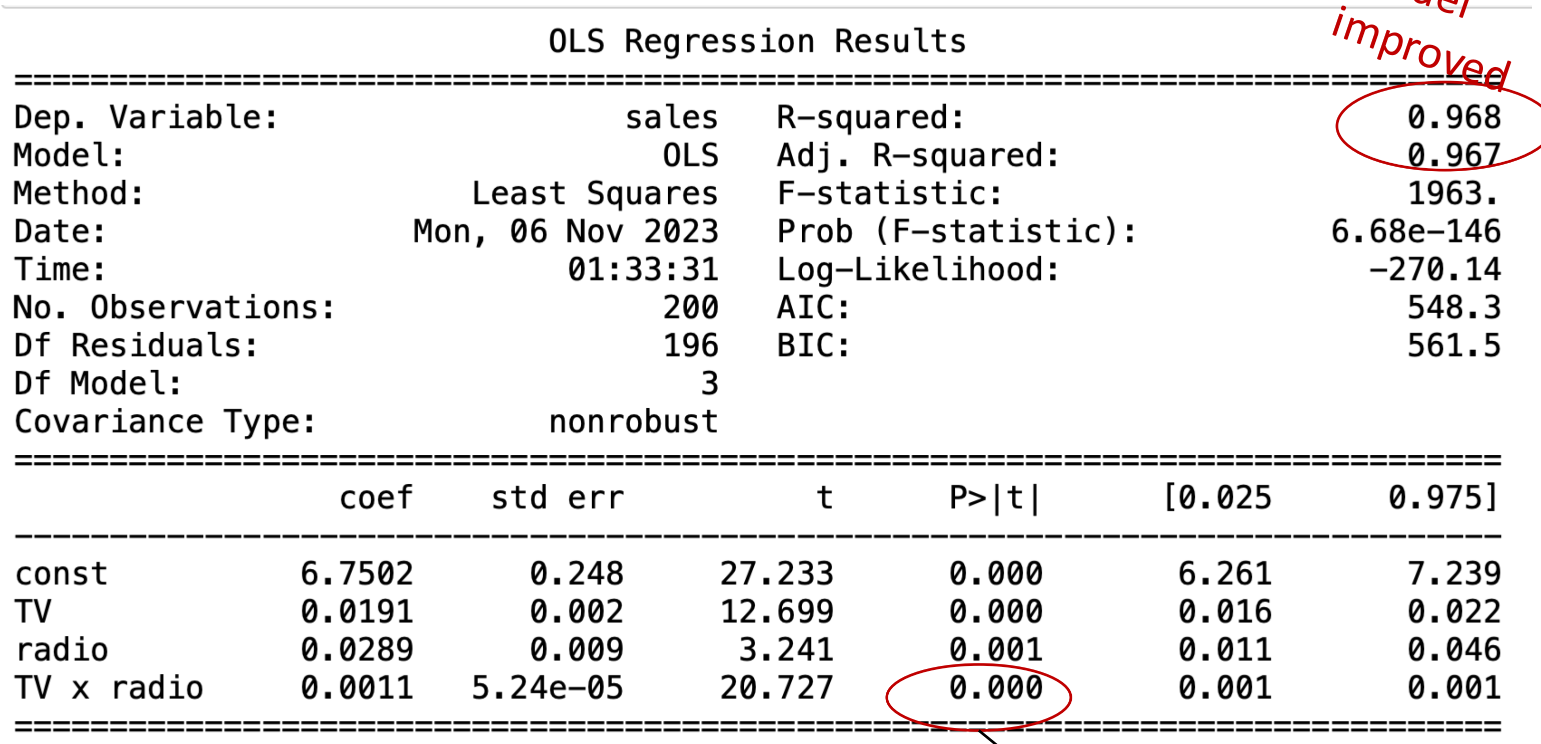
- The new model with the interaction: **sales** = β_0 + β_1 **TV** + β_2 **radio** + β_3 **TV** × **radio**

new model

sales = β_0 + β_1 **TV** + β_2 **radio** + β_3 **TV** × **radio**



underestimation fixed! 😃



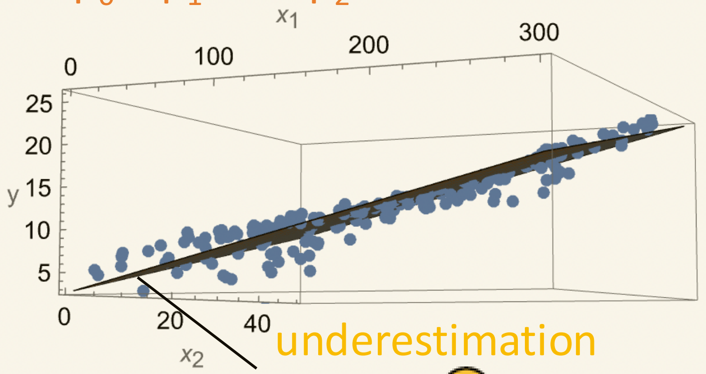
model improved

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  sales   R-squared:                       0.968
Model:                            OLS   Adj. R-squared:                  0.967
Method:                 Least Squares   F-statistic:                     1963.
Date:                Mon, 06 Nov 2023   Prob (F-statistic):          6.68e-146
Time:                        01:33:31   Log-Likelihood:                -270.14
No. Observations:                 200   AIC:                             548.3
Df Residuals:                     196   BIC:                             561.5
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          6.7502      0.248     27.233      0.000       6.261       7.239
TV             0.0191      0.002     12.699      0.000       0.016       0.022
radio          0.0289      0.009      3.241      0.001       0.011       0.046
TV x radio     0.0011   5.24e-05     20.727      0.000       0.001       0.001
==============================================================================
```

TV x radio term is important

old model:

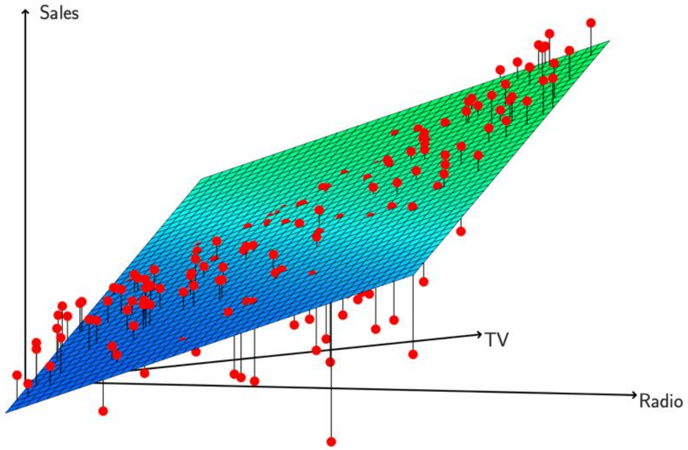
sales = β_0 + β_1 **TV** + β_2 **radio**



underestimation present 😔

Linear Regression: Interaction terms

- Interaction (synergetic effect of two or more variables) is about the correlation between the response variable y and the interaction term $\mathbf{x_1} \times \mathbf{x_2}$. So if the **TV** × **radio** term affects the **sales**, we need to add this term to the model. We can visually verify if we need an interaction term: Look at the 3D surface plot of **sales** vs **TV** and **radio** and check if the corners in the data are curved or just planar.
- Interpretation: **TV** ads and **radio** ads increase sales. This data also shows that when **radio** and **TV** ads used together, the effect on **sales** is more than "**radio** ad effect" + **"TV** ad effect."



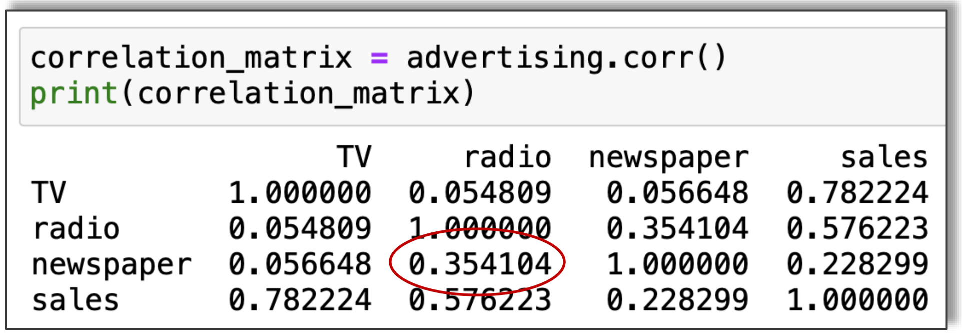
- Another example for interaction:

Ali is hardworking, Veli is hardworking too, but when Ali and Veli are assigned to work together, they chat a lot and do less work than they would do when they're alone. So, you need –**Ali's work** × **Veli's work** term for the response variable **total_work_done** in the model.

Linear Regression: (Multi)Collinearity

- Collinearity is about the correlation between the predictors such as correlation between **TV** and **radio** or between **newspaper** and **radio**. Look at the correlation matrix or scatter plots between the predictors to see if there is correlation between predictors.

Correlation matrix



```
correlation_matrix = advertising.corr()
print(correlation_matrix)
```

```
                 TV     radio  newspaper     sales
TV         1.000000  0.054809   0.056648  0.782224
radio      0.054809  1.000000   0.354104  0.576223
newspaper  0.056648  0.354104   1.000000  0.228299
sales      0.782224  0.576223   0.228299  1.000000
```

- For linear regression, multicollinearity is a big problem: it makes the model unreliable. Model can't decide which predictor to assign the effect on the target.
- Example for collinearity :

Ali does real work but Veli does not do any real work except solely imitating Ali. This is correlation between the features **Ali's work** and **Veli's work.** Existence of correlation does not necessitate of an interaction term in the model. We don't necessarily need to add "**Ali's_work** × **Veli's_work**" interaction term just because of the correlation. (If there is interaction effects in the data, we can, but not because of correlation.)

- Collinearity and interaction are independent concepts. Don't conflate these two!

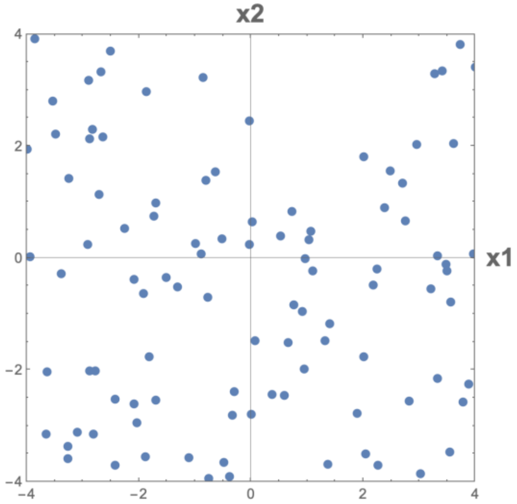
Linear Regression: (Multi)Collinearity vs interaction

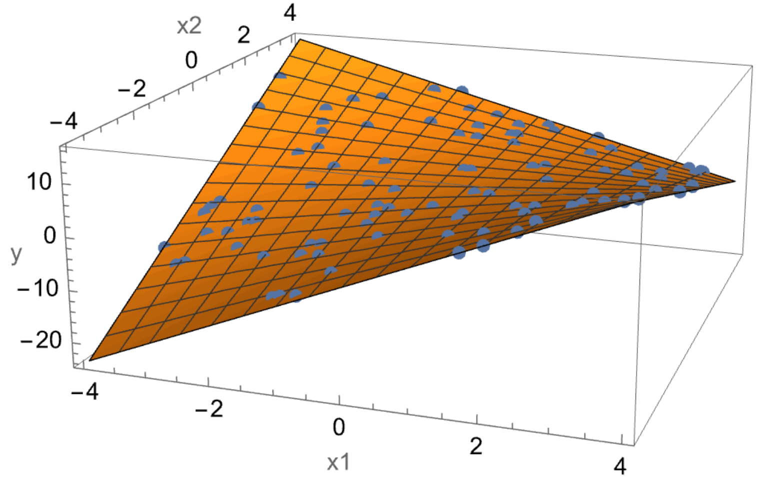
- <u>Scatter plot of features:</u>

$\mathbf{x_1}$ and $\mathbf{x_2}$ are *not* correlated.

- <u>3D plot of $\mathbf{x_1}$, $\mathbf{x_2}$ and **y**:</u>

Interaction term $\mathbf{x_1} \times \mathbf{x_2}$ is needed because data doesn’t fall on the plane but it curves.



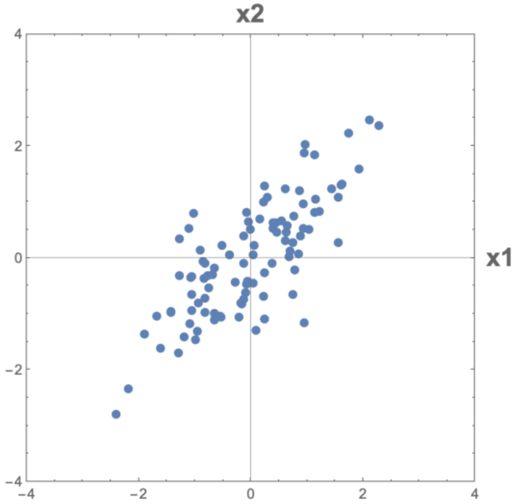


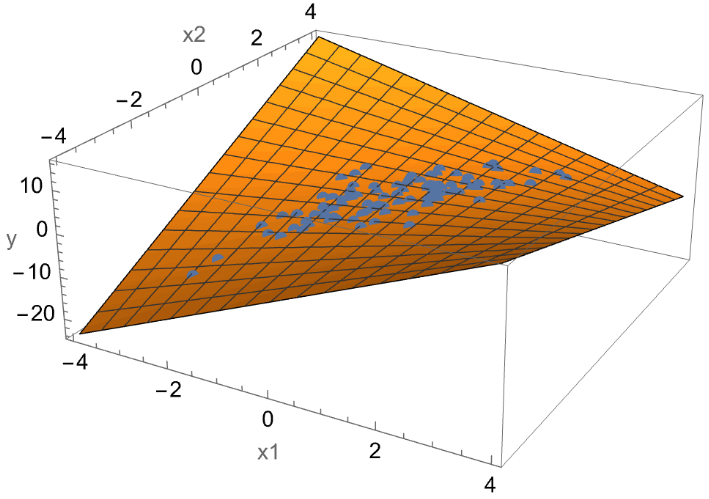
- <u>Scatter plot of features:</u>

$\mathbf{x_1}$ and $\mathbf{x_2}$ are correlated.

- <u>3D plot of $\mathbf{x_1}$, $\mathbf{x_2}$ and **y**:</u>

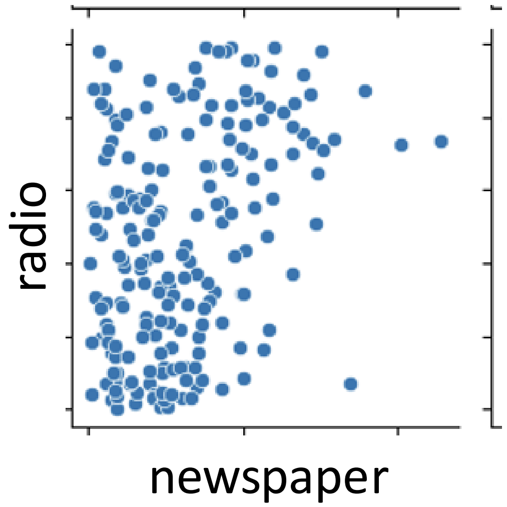
Interaction term $\mathbf{x_1} \times \mathbf{x_2}$ is needed because data doesn’t fall on the plane but it curves





Linear Regression: Curing (multi)collinearity

- Collinearity can be observed from the correlation matrix. Remember **newspaper** and **radio** has some correlation (0.35). Using both predictors yielded worse R^2.
- The easiest cure for collinearity is removing one varible. Which one? Try removing one of them and choose the model that gives rise to the highest R^2. (If you will use the model for prediction, perform model selection by using train-validation-test split and calculate error metrics MSE/RMSE/MAE/MAPE).
- There are other methods to cure collinearity like Principal Component Regression which combines the correlated predictors into single new predictor. For example,



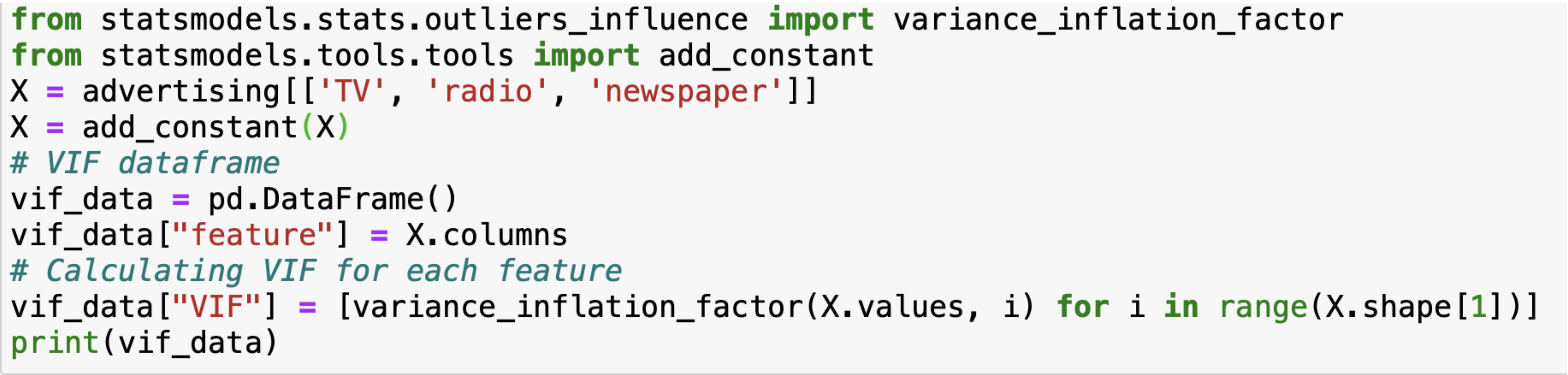
$$\textbf{newpredictor} = \frac{1}{\sqrt{2}}\,\textbf{newspaper} + \frac{1}{\sqrt{2}}\,\textbf{radio}$$

so the new model becomes

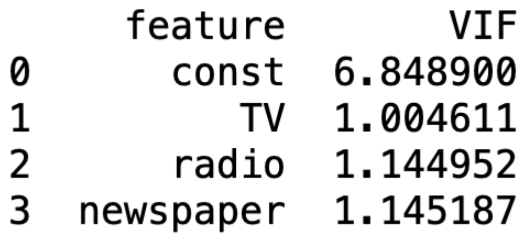
$$\textbf{sales} = \beta_0 + \beta_1\,\textbf{TV} + \beta_2\,\textbf{newpredictor}$$

Linear Regression: Curing collinearity

- There may be correlation between the predictors, but it does mean this correlation will always ruin our regression model.
- Best way to check if a predictor has multicollinearity with other predictors is calculating the Variance Inflation Factor (VIF). If VIF is larger than 5-10, then we can consider removing that predictor. Minimum of VIF is 1.
- Multicollinearity is usually a problem when correlation between features > 0.7.



```
from statsmodels.stats.outliers_influence import variance_inflation_factor
from statsmodels.tools.tools import add_constant
X = advertising[['TV', 'radio', 'newspaper']]
X = add_constant(X)
# VIF dataframe
vif_data = pd.DataFrame()
vif_data["feature"] = X.columns
# Calculating VIF for each feature
vif_data["VIF"] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]
print(vif_data)
```



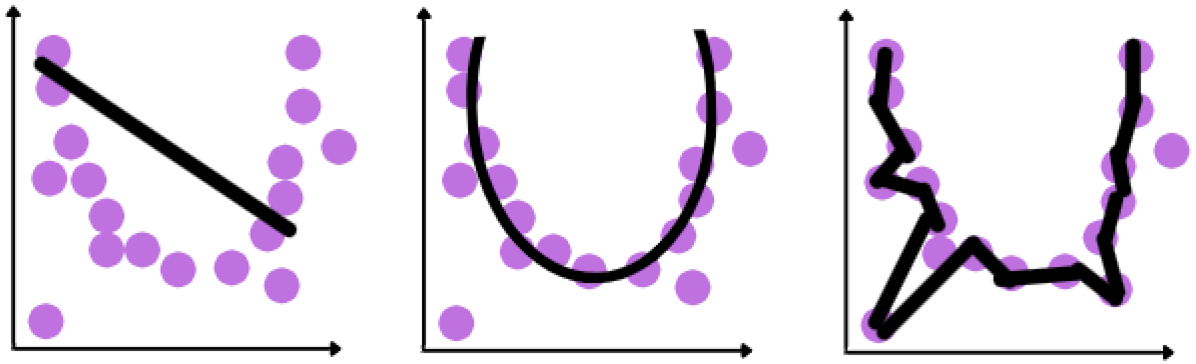
```
     feature       VIF
0      const  6.848900
1         TV  1.004611
2      radio  1.144952
3  newspaper  1.145187
```

(Ignore VIF of const, which is β_0)

(Looks like there isn't a significant multicollinearity between the predictors. But we've seen before that removing **newspaper** improved the model. Yes, but apparently this improvement wasn't due to only removing multicollinearity. **newspaper** is simply not a good predictor.)

Regularization

- Regularization
  i. Avoids overfitting so increases model predictive power,
  ii. Cures multicollinearity.



Underfitting

(model is too simple)

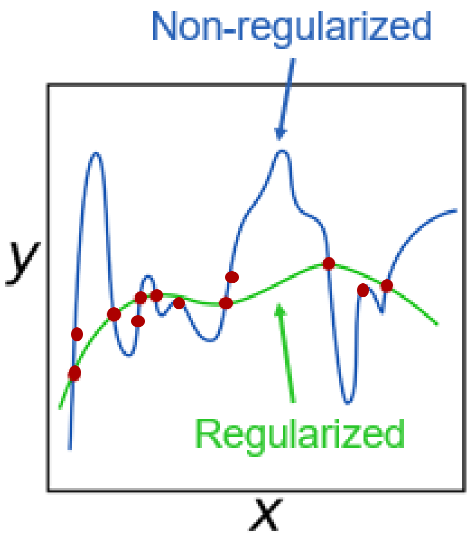
Optimal

Overfitting

(model is too complex and captures even noise in the data)

- Why is there a generalization gap between training and test data?
  a) Overfitting (model learns statistical peculiarities/fluctuations/flukes).
  b) Model unconstrained in areas where there are no training examples.
- Regularization = methods to reduce the generalization gap;

the gap between train error and test error.



Ridge and Lasso Regularization

Ridge (L2 regularization)

- Shrinks β's evenly but does not typically make them exactly zero. Prevents some β's becoming very large so model doesn't become very complex and overfit.

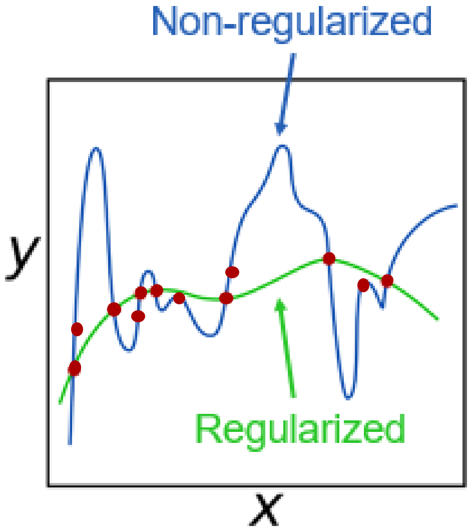
$$Loss = \sum_{i=1}^{N} (\hat{y}_i - y_i)^2 + \lambda \sum_{i=1}^{p} \beta_i^2$$

Lasso (L1 regularization)

- Can shrink less important β's to exactly zero, eliminating bad predictors from the model. So it also acts as feature selection.

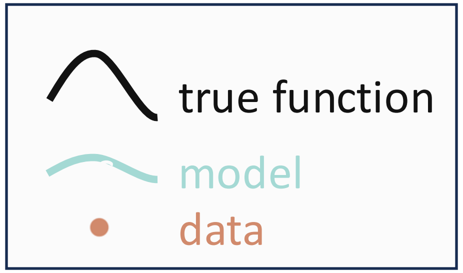
$$Loss = \sum_{i=1}^{N} (\hat{y}_i - y_i)^2 + \lambda \sum_{i=1}^{p} |\beta_i|$$

- λ is hyperparameter between 10^{-3} and 10^{+3}.
- Regularization encourages smaller β's (=slopes). Smaller β's, less wiggle in model (consider polynomial regression).
- One practical advice: It's best to use lasso during exploration/experiment stage and select the features that matter the most. Once you settle on the features, ridge provides better generalizable models (or try ridge and lasso and take the best).
- You may need to standardize or normalize your features before training.

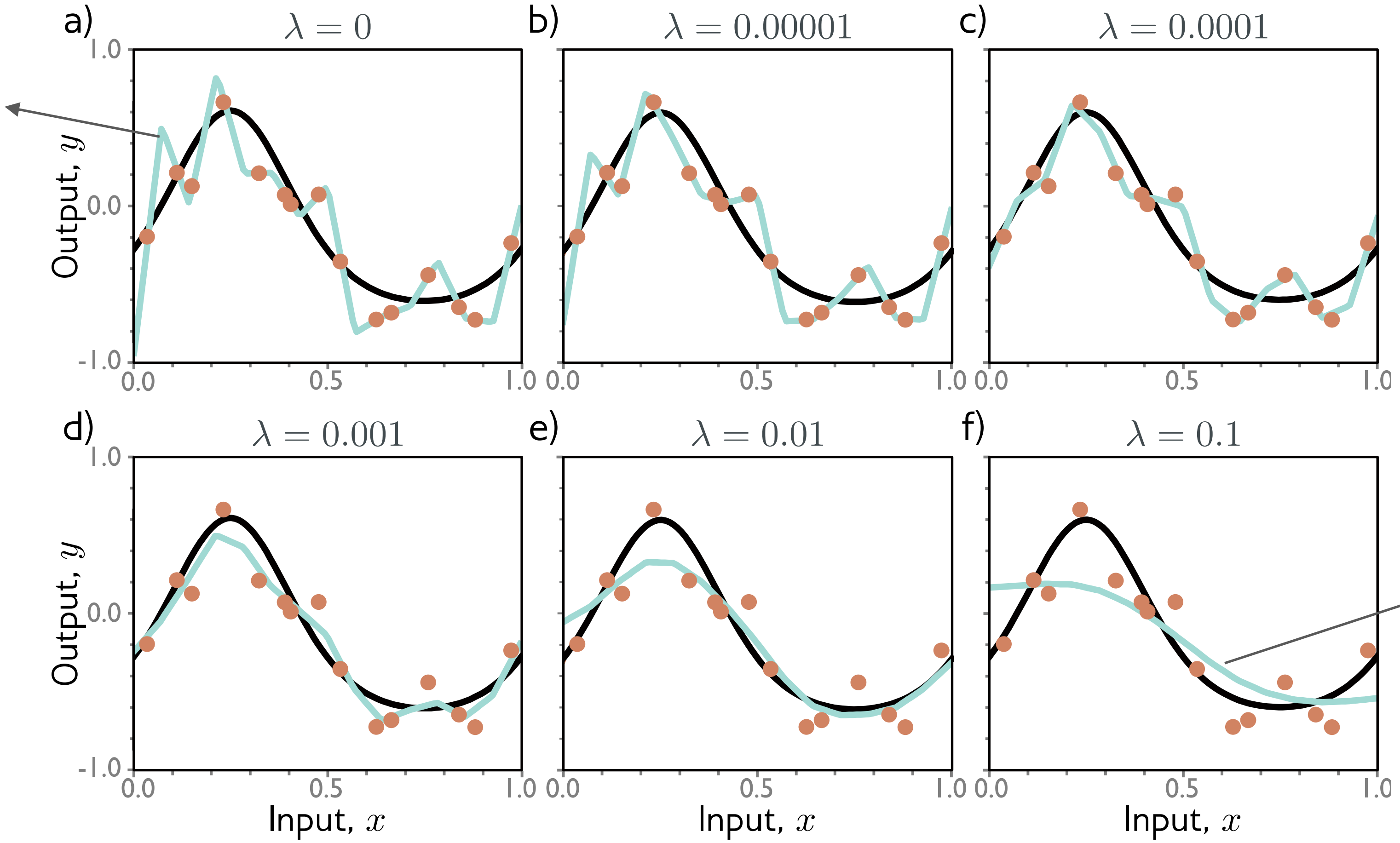


Ridge and Lasso Regularization

- λ is hyperparameter between 10^{-3} and 10^{+3}.



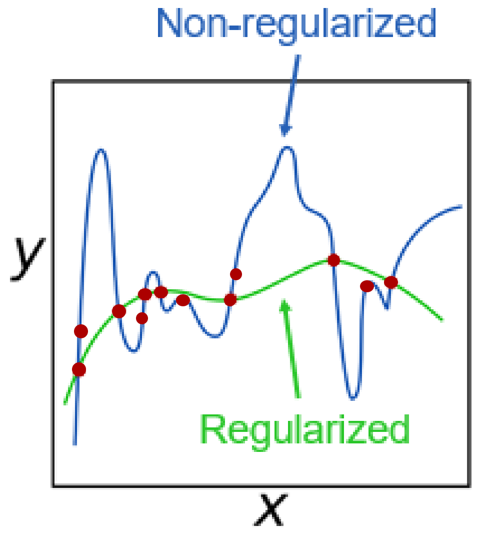
The model is way too <span style="color:red">overfitted</span> due to high model complexity.



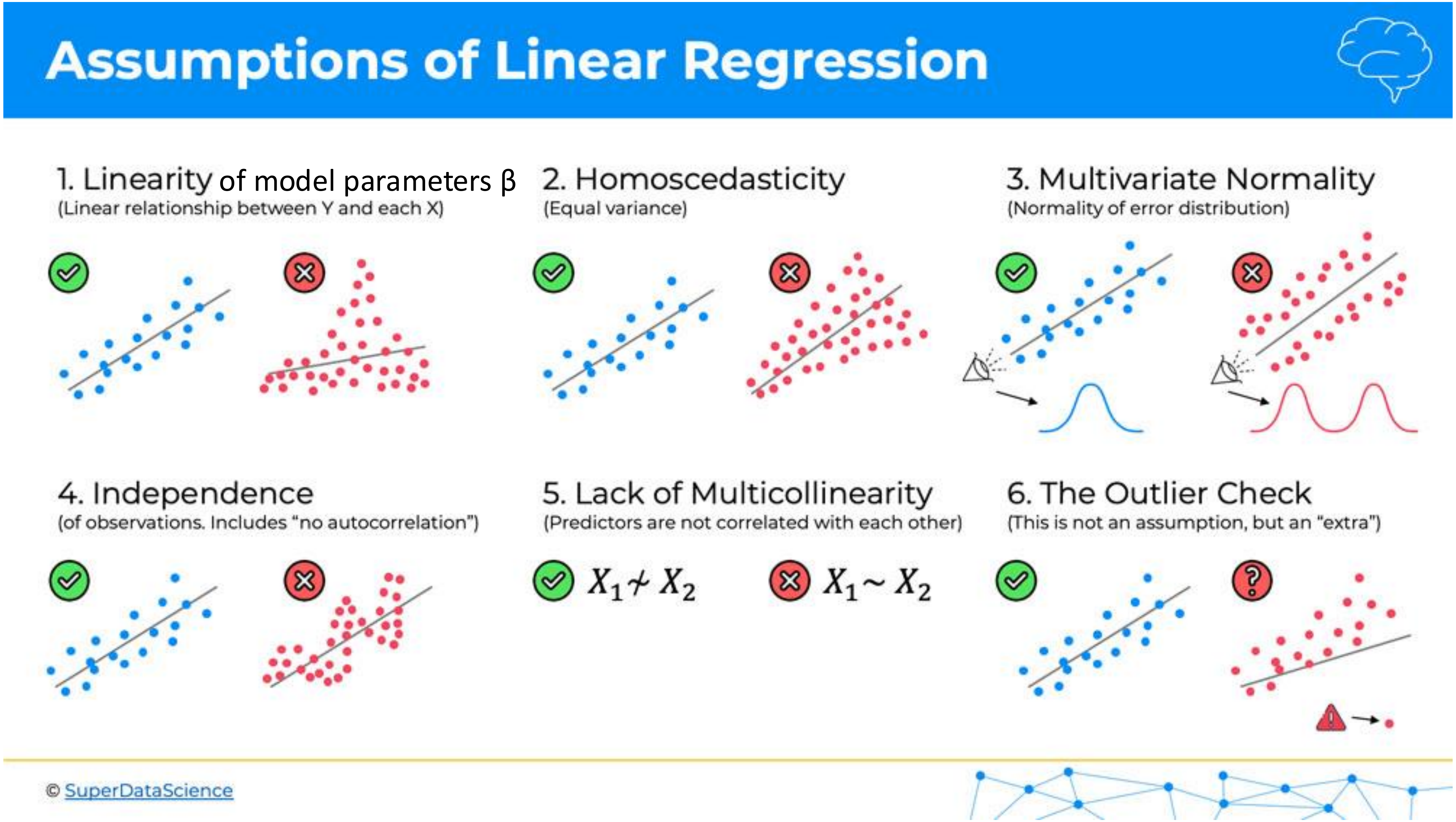
The model is way too smooth.

Why does ridge/lasso regularization help?

- As λ increases, the model becomes smoother, the fit to the data becomes less accurate (more bias). Helps improve test performance because
  a) Prevents overfitting since it allows less curvature in the model so the model cannot try to pass close to every data points,
  b) When model is too complex, it curves a lot at places in the feature space where data is scarce or missing. Ridge/lasso regularization makes the model interpolate with smoother functions, i.e., not wiggle a lot.
- Discourages slavish adherence to the data (overfitting).
- Encourages smoothness between datapoints.



Linear Regression: Conditions



Assumptions of Linear Regression

1. Linearity of model parameters β
(Linear relationship between Y and each X)

✓ ✗

2. Homoscedasticity
(Equal variance)

✓ ✗

3. Multivariate Normality
(Normality of error distribution)

✓ ✗

4. Independence
(of observations. Includes "no autocorrelation")

✓ ✗

5. Lack of Multicollinearity
(Predictors are not correlated with each other)

✓ $X_1 \nsim X_2$ ✗ $X_1 \sim X_2$

6. The Outlier Check
(This is not an assumption, but an "extra")

✓ ?

© SuperDataScience

Linear Regression: Residuals

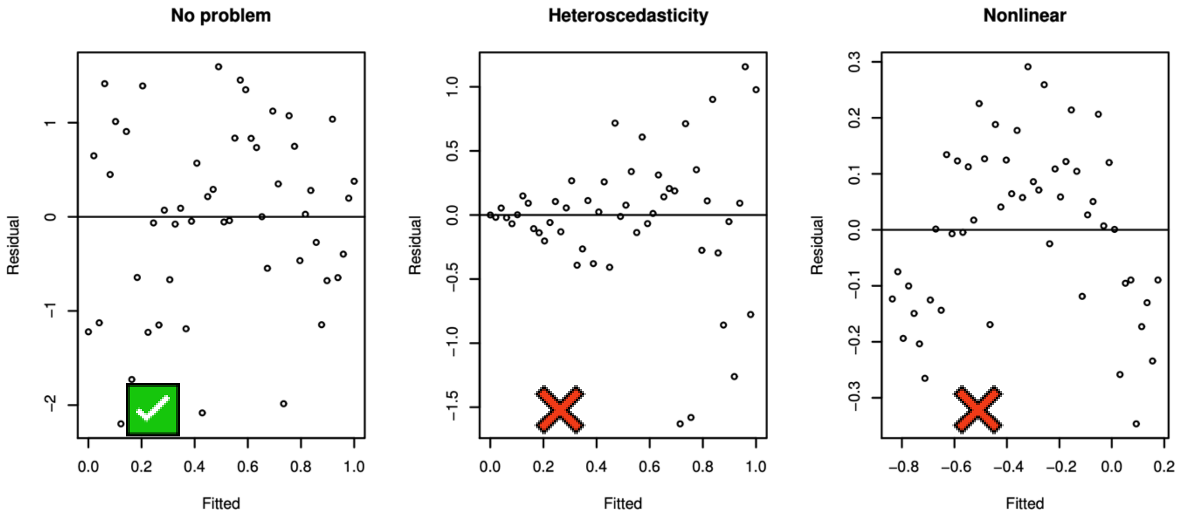
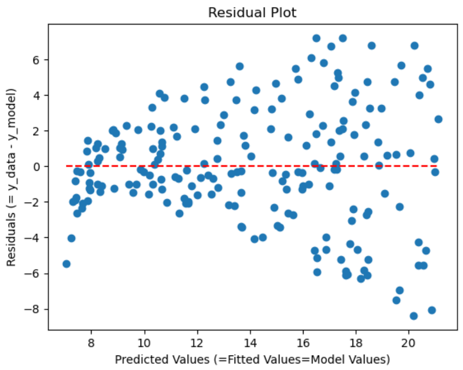


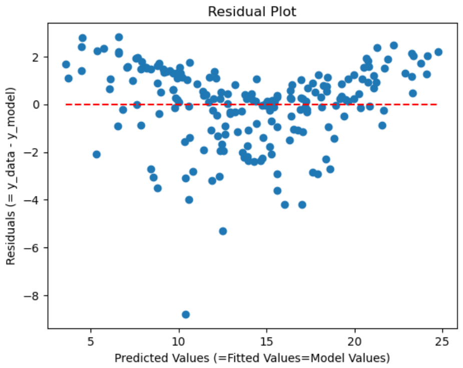
Figure 6.1 Residuals vs. fitted plots — the first suggests no change to the current model while the second shows nonconstant variance and the third indicates some nonlinearity, which should prompt some change in the structural form of the model.

Model: **sales ~ TV**



Problem: strong heteroscedasticity, i.e., loss of homoscedasticity

Model: **sales ~ TV + radio**

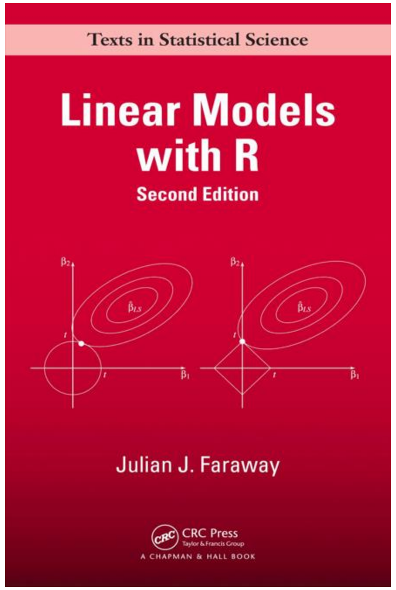


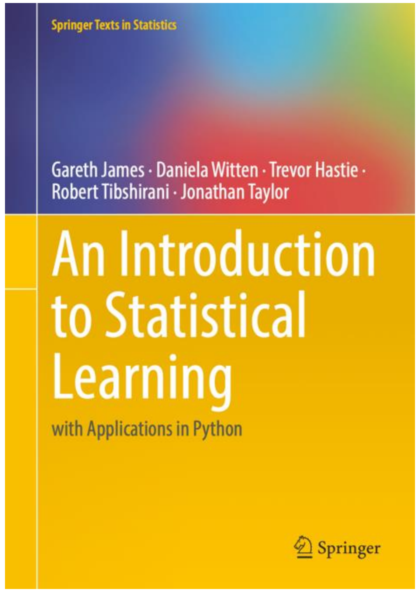
Problem: Residuals (error) depends on fitted y values

→ Jupyter notebook.

Linear Regression: Final words

- One whole semester is needed to cover linear regression thoroughly. We'll stop here. When you want to do serious work, you may want to check the methods and tests used in these books:




Springer Texts in Statistics
Gareth James · Daniela Witten · Trevor Hastie · Robert Tibshirani · Jonathan Taylor
An Introduction to Statistical Learning
with Applications in Python
Springer

- Don't forget: The computer will always give you a regression line/curve, but it may not be sensible.
- Note: You can do regression with decision trees, neural networks etc.