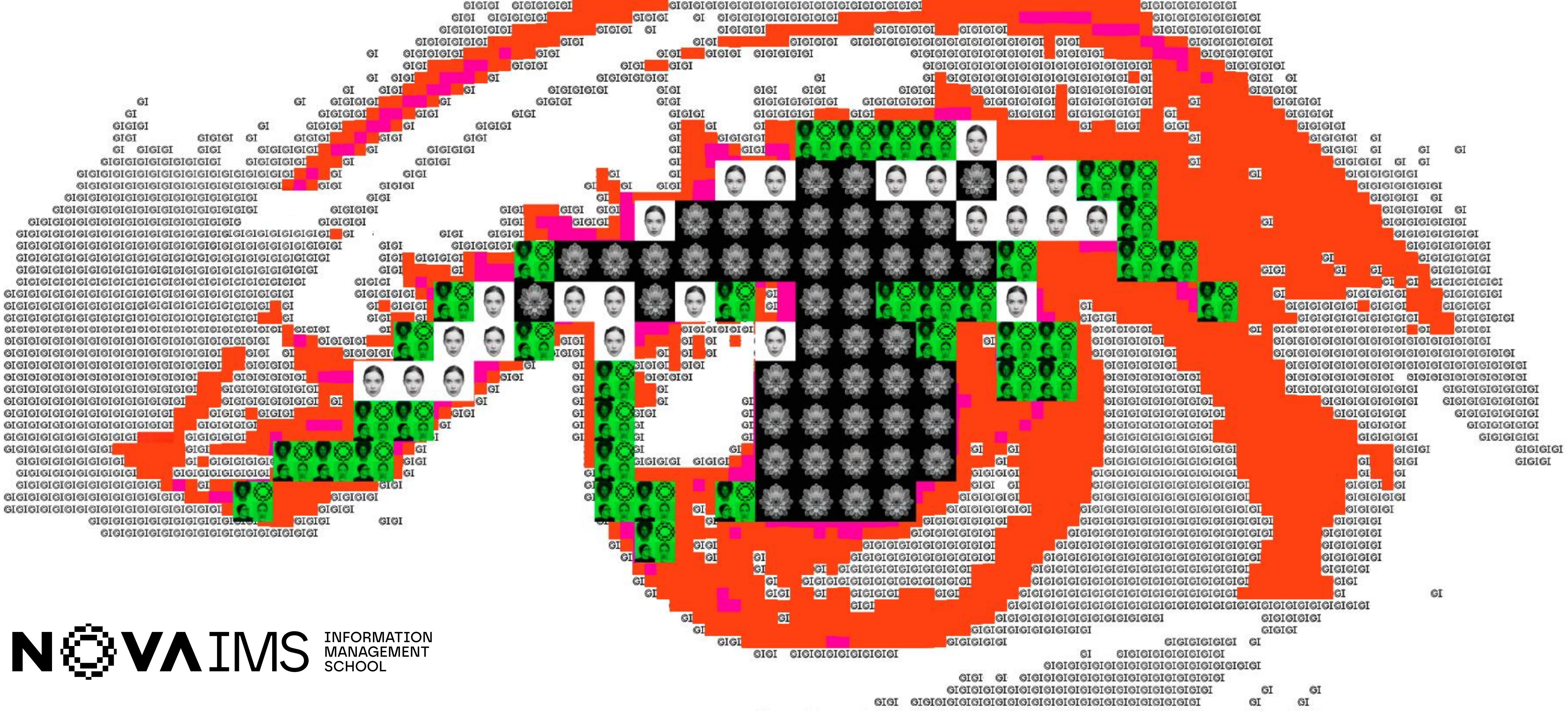
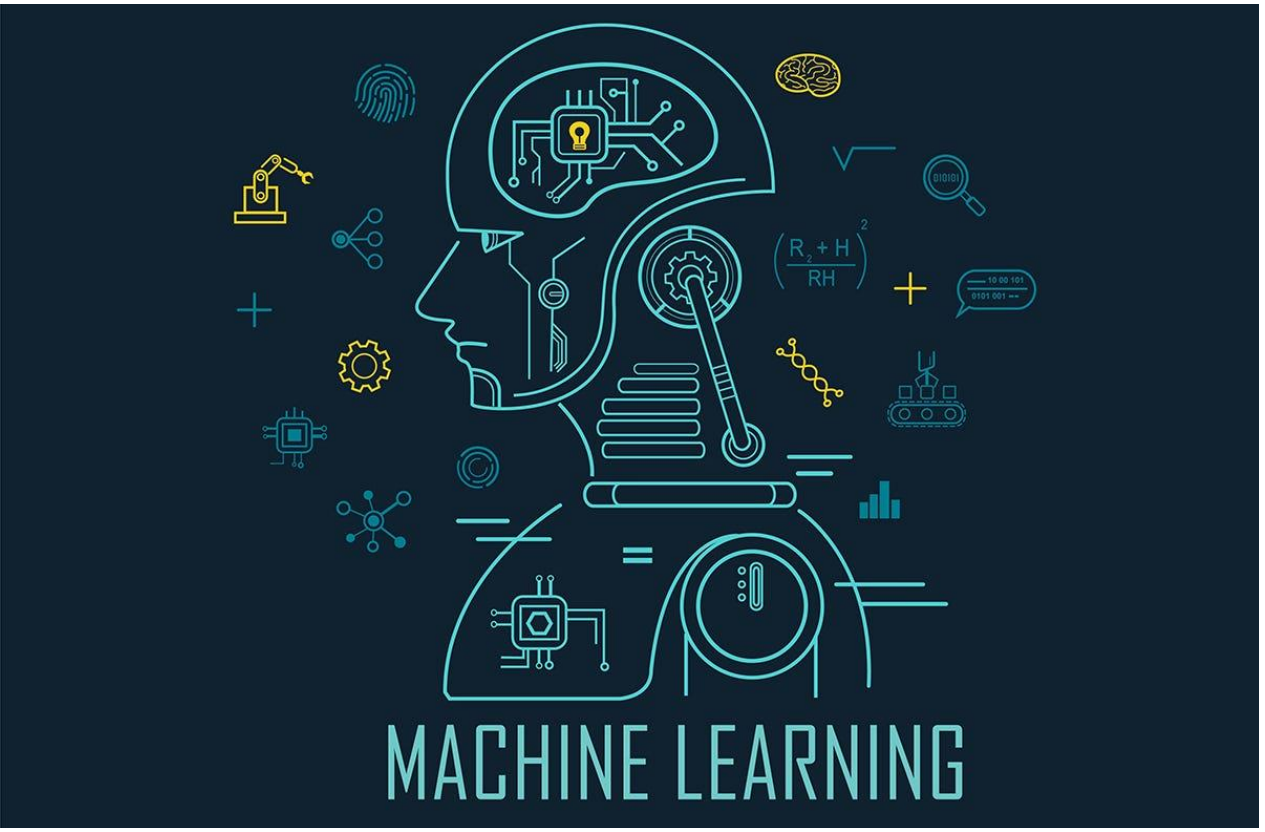
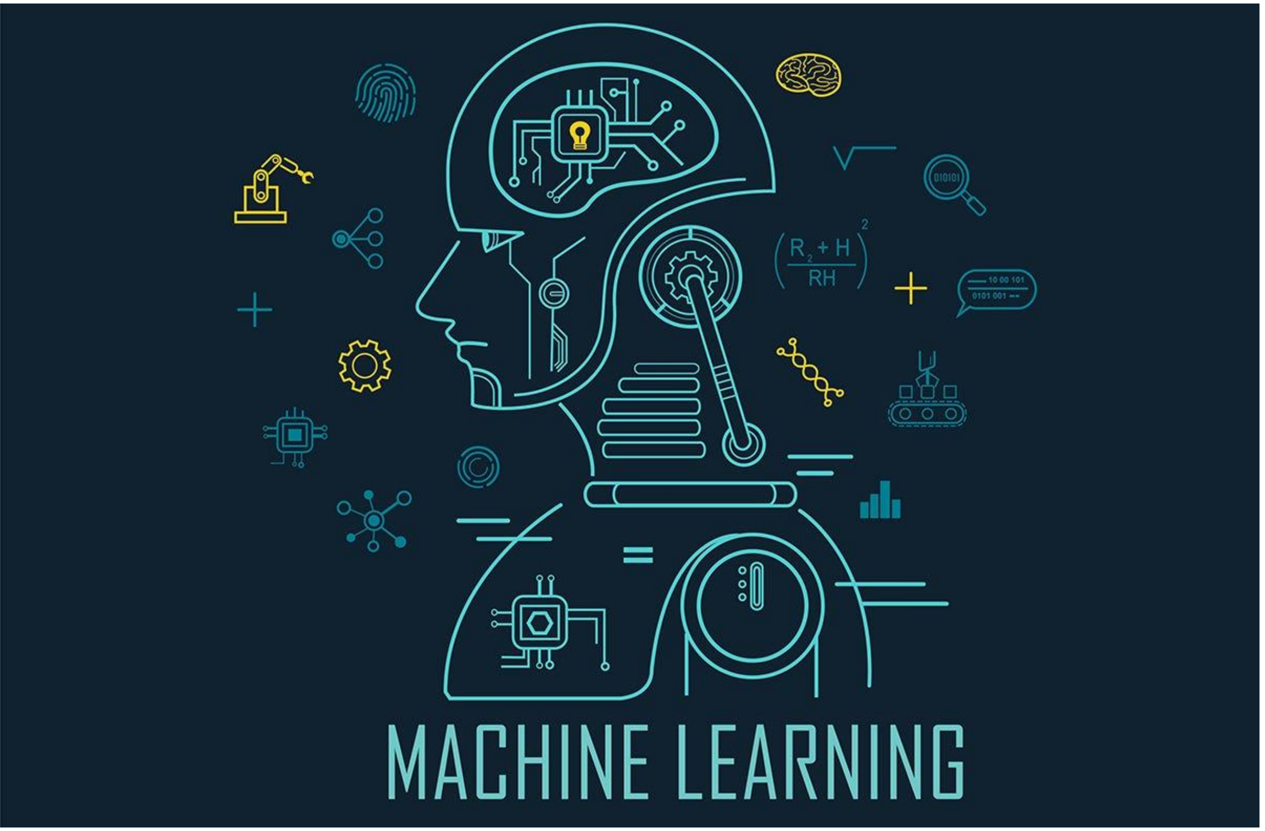


Machine Learning in Finance

- Prof. Ian J. Scott

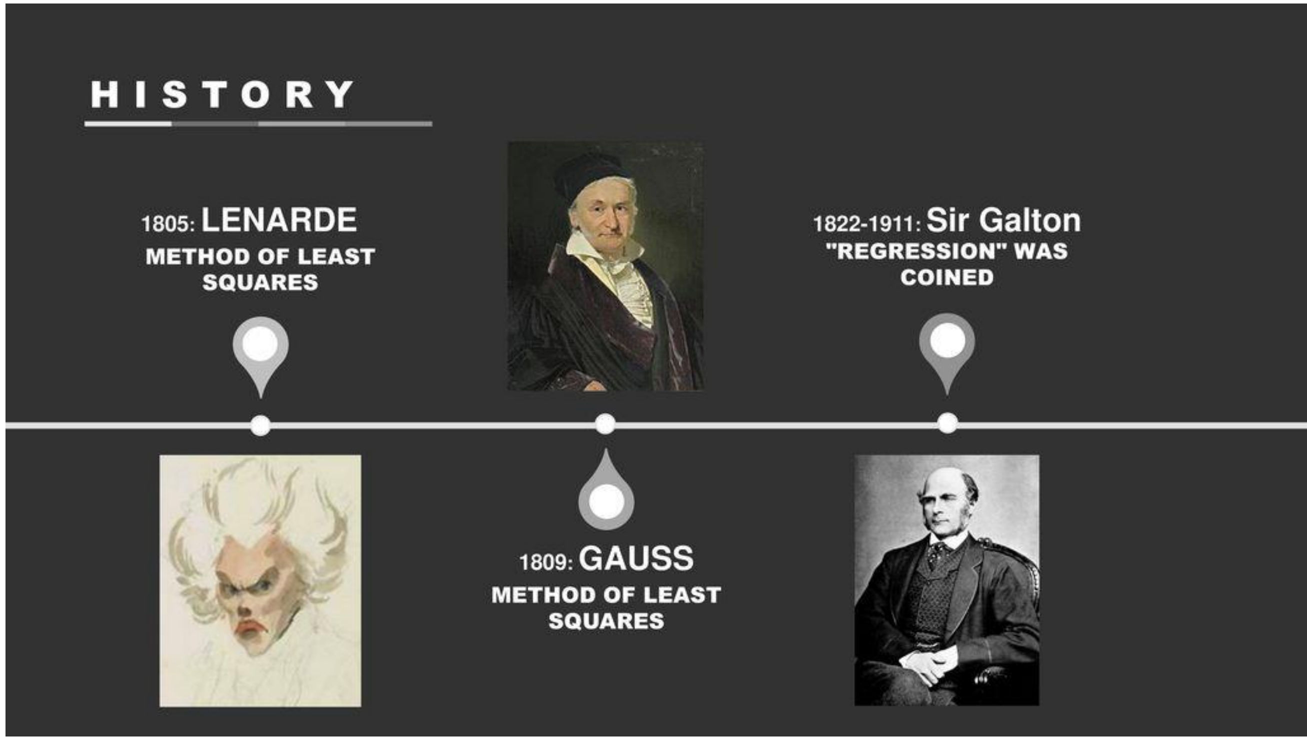


Project Online

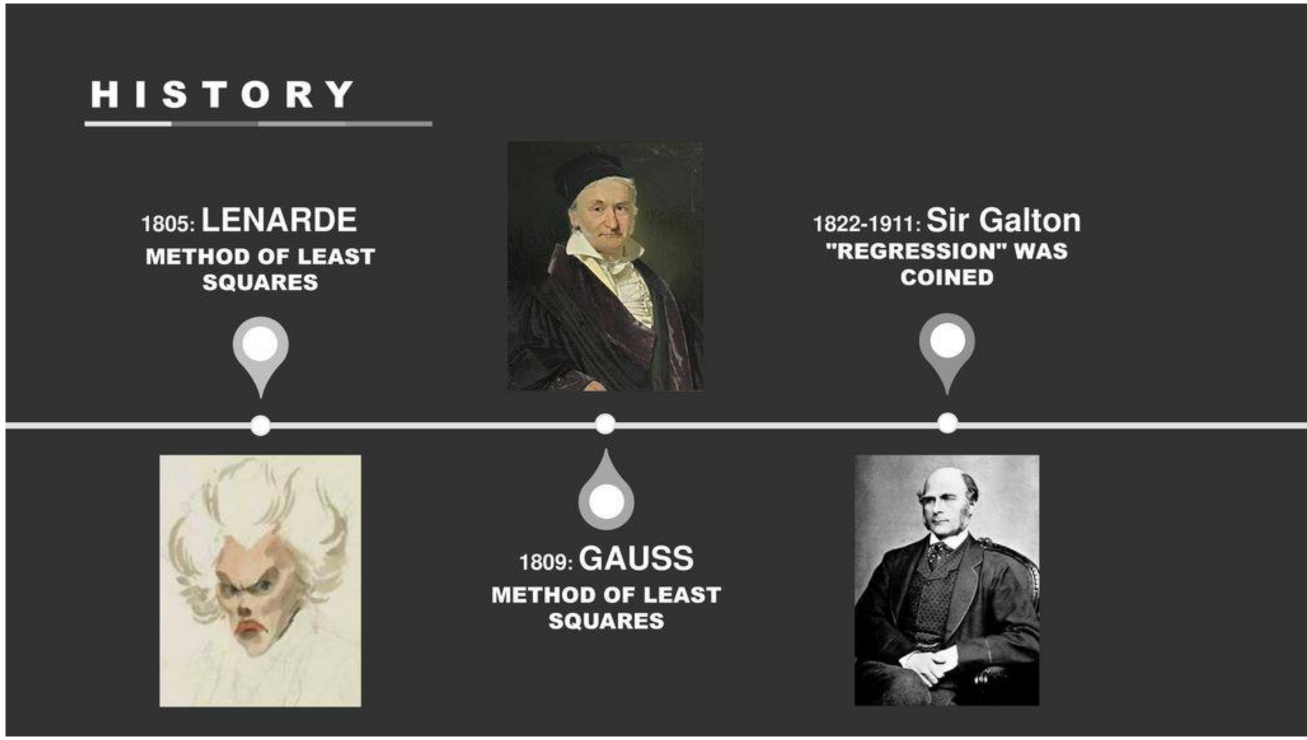


Regularisation

- The linear model has distinct advantages in terms of inference and, on real world problems, is often surprisingly competitive in relation to non-linear methods.



- We are going to look at ways in which the simple linear model can be improved, **especially for more complex datasets**, by replacing plain least squares fitting with some alternative fitting procedures.
- These technique we will use, **regularisation**, is also commonly applied in other situations.

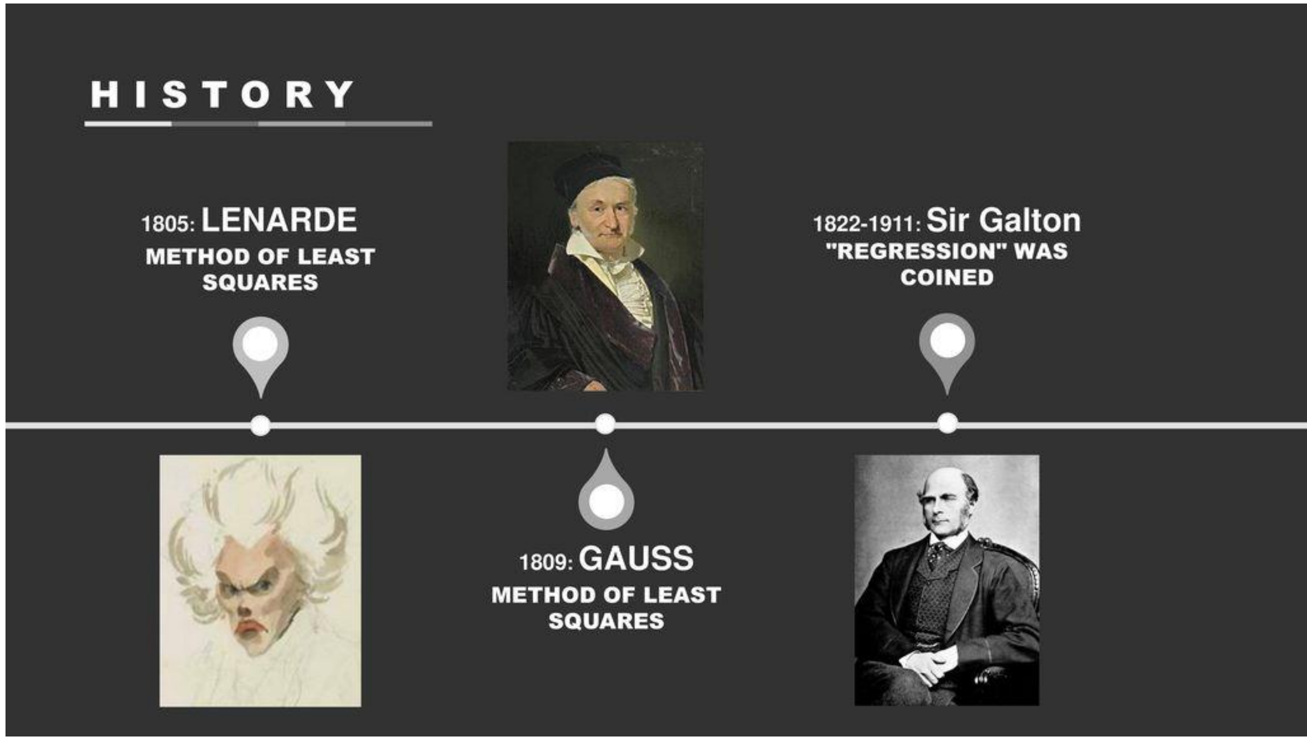


Regularisation Disadvantages:

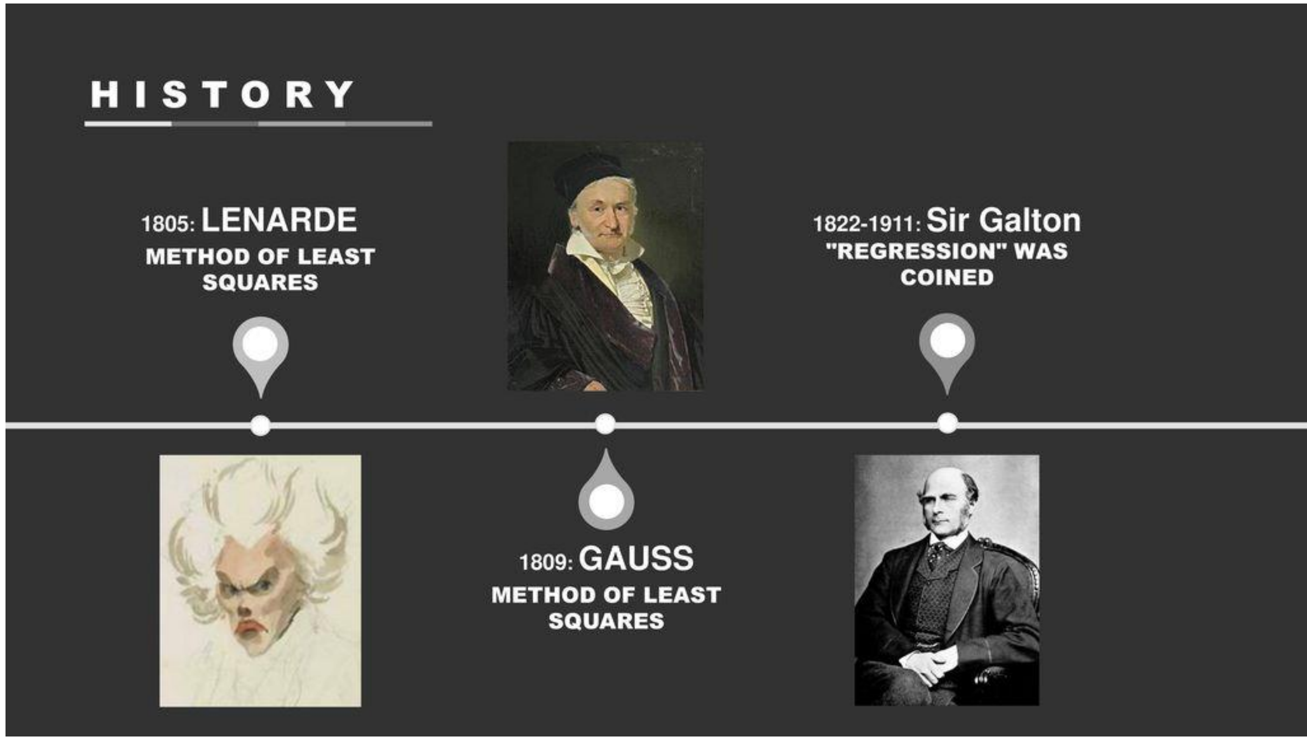
- Estimator no longer BLUE.

Regularisation Advantages:

- **Prediction accuracy** as the ratio of p to n grows (OLS has problems when p > n as there is not a unique solution).
- **Model Interpretability** as we help identify the most important features.



- **Subset Selection.** This approach involves identifying a subset of the p predictors that we believe to be related to the response.
- **Shrinkage.** This approach involves fitting a model involving all p predictors. However, the estimated coefficients are shrunken towards zero relative to the least squares estimates. This shrinkage (also known as regularization) has the effect of reducing variance. Similar to the Bayesian regression we saw with 0 as a prior.
- **Dimension Reduction.** This approach involves projecting the p predictors into an M-dimensional subspace, where M < p. This is achieved by computing M different linear combinations, or projections, of the variables.



Algorithm 6.1 *Best subset selection*

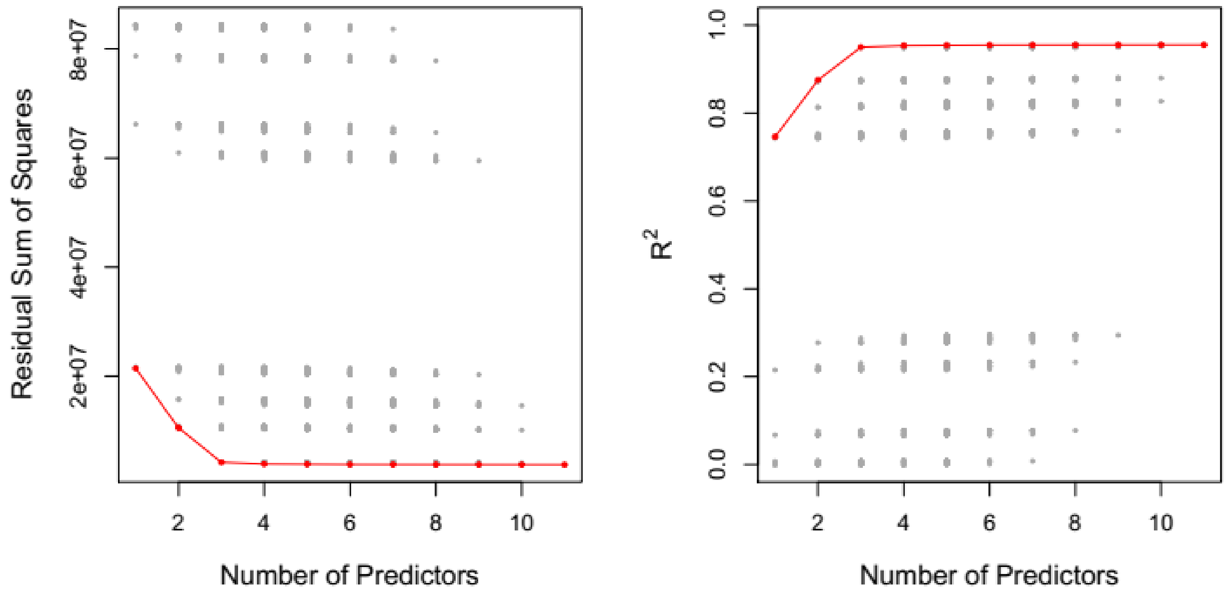
```
1. Let $\mathcal{M}_0$ denote the *null model*, which contains no predictors. This
   model simply predicts the sample mean for each observation.

2. For $k = 1, 2, \ldots p$:

   (a) Fit all $\binom{p}{k}$ models that contain exactly $k$ predictors.

   (b) Pick the best among these $\binom{p}{k}$ models, and call it $\mathcal{M}_k$. Here *best*
       is defined as having the smallest RSS, or equivalently largest $R^2$.

3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using using the
   prediction error on a validation set, $C_p$ (AIC), BIC, or adjusted $R^2$.
   Or use the cross-validation method.
```



Algorithm 6.2 *Forward stepwise selection*

```
1. Let $\mathcal{M}_0$ denote the *null* model, which contains no predictors.
2. For $k = 0, \ldots, p-1$:
    (a) Consider all $p - k$ models that augment the predictors in $\mathcal{M}_k$
        with one additional predictor.
    (b) Choose the *best* among these $p - k$ models, and call it $\mathcal{M}_{k+1}$.
        Here *best* is defined as having smallest RSS or highest $R^2$.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using the pre-
   diction error on a validation set, $C_p$ (AIC), BIC, or adjusted $R^2$. Or
   use the cross-validation method.
```

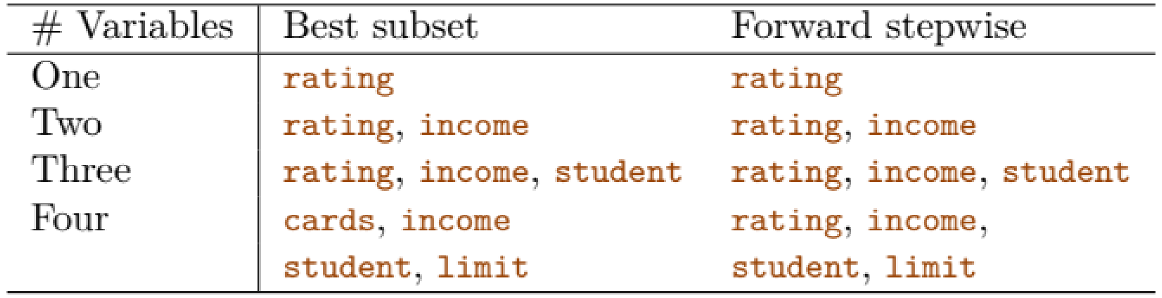
# Variables	Best subset	Forward stepwise
One	`rating`	`rating`
Two	`rating`, `income`	`rating`, `income`
Three	`rating`, `income`, `student`	`rating`, `income`, `student`
Four	`cards`, `income`, `student`, `limit`	`rating`, `income`, `student`, `limit`

Algorithm 6.3 *Backward stepwise selection*

```
1. Let $\mathcal{M}_p$ denote the *full* model, which contains all $p$ predictors.
2. For $k = p, p-1, \ldots, 1$:
   (a) Consider all $k$ models that contain all but one of the predictors
       in $\mathcal{M}_k$, for a total of $k-1$ predictors.
   (b) Choose the *best* among these $k$ models, and call it $\mathcal{M}_{k-1}$. Here
       *best* is defined as having smallest RSS or highest $R^2$.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using the prediction
   error on a validation set, $C_p$ (AIC), BIC, or adjusted $R^2$. Or
   use the cross-validation method.
```

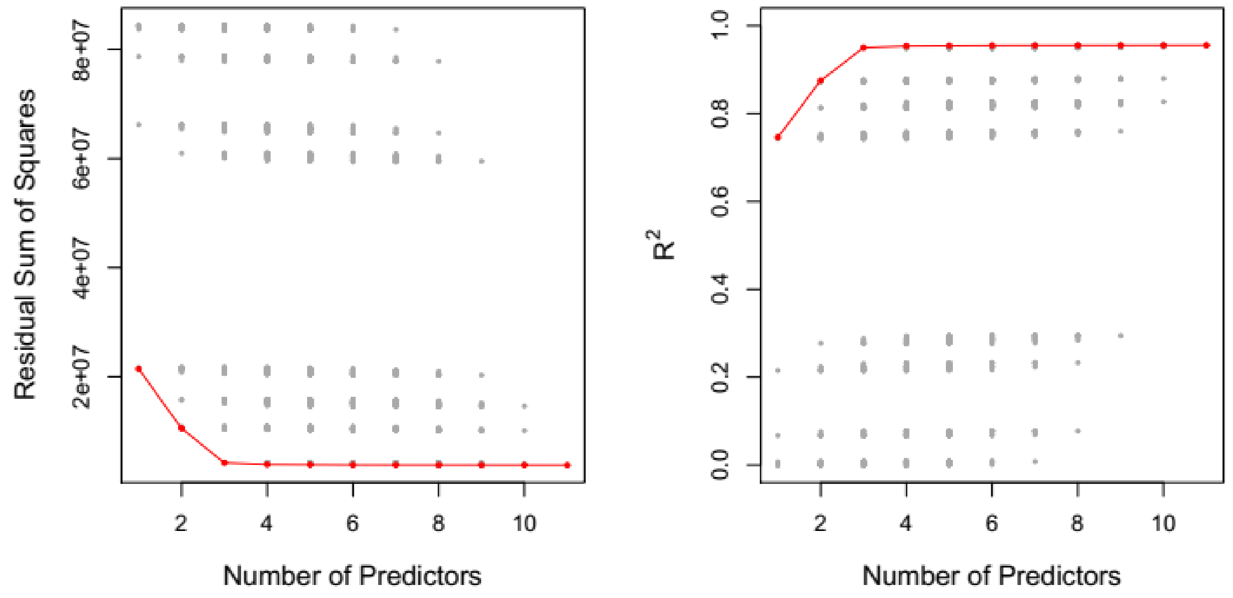
# Variables	Best subset	Forward stepwise
One	`rating`	`rating`
Two	`rating`, `income`	`rating`, `income`
Three	`rating`, `income`, `student`	`rating`, `income`, `student`
Four	`cards`, `income`, `student`, `limit`	`rating`, `income`, `student`, `limit`

- Hybrid



# Variables	Best subset	Forward stepwise
One	`rating`	`rating`
Two	`rating`, `income`	`rating`, `income`
Three	`rating`, `income`, `student`	`rating`, `income`, `student`
Four	`cards`, `income` `student`, `limit`	`rating`, `income`, `student`, `limit`

- How do we choose?
- MAE, RMSE, R2 typically improve with more features
- But we risk overfitting!

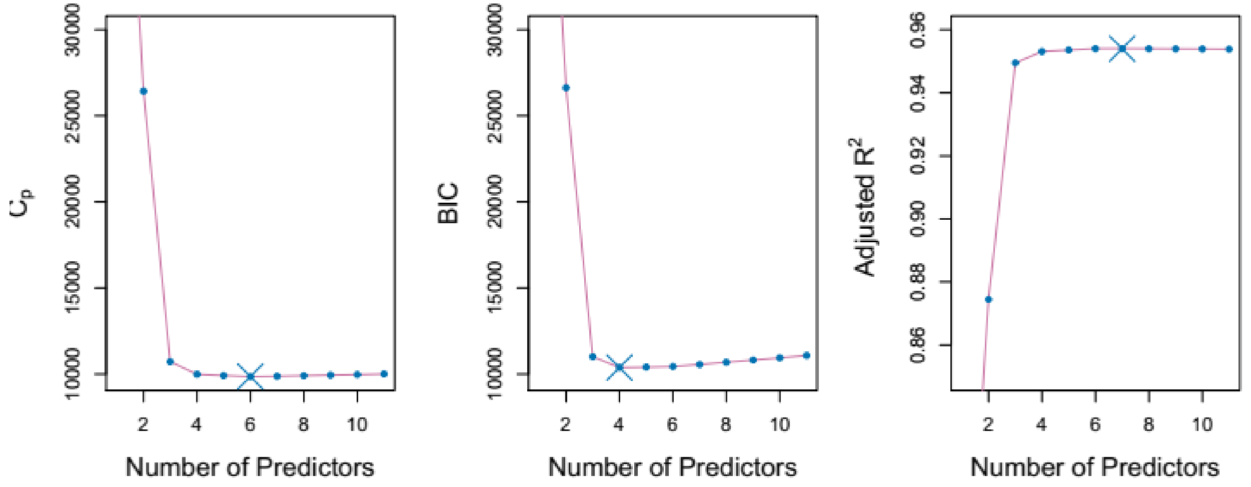


Linear

- **Adjusted R2** - More interpretable than some other metrics since it's in terms of the variability explained.
- **Cp** - Focuses on prediction error.

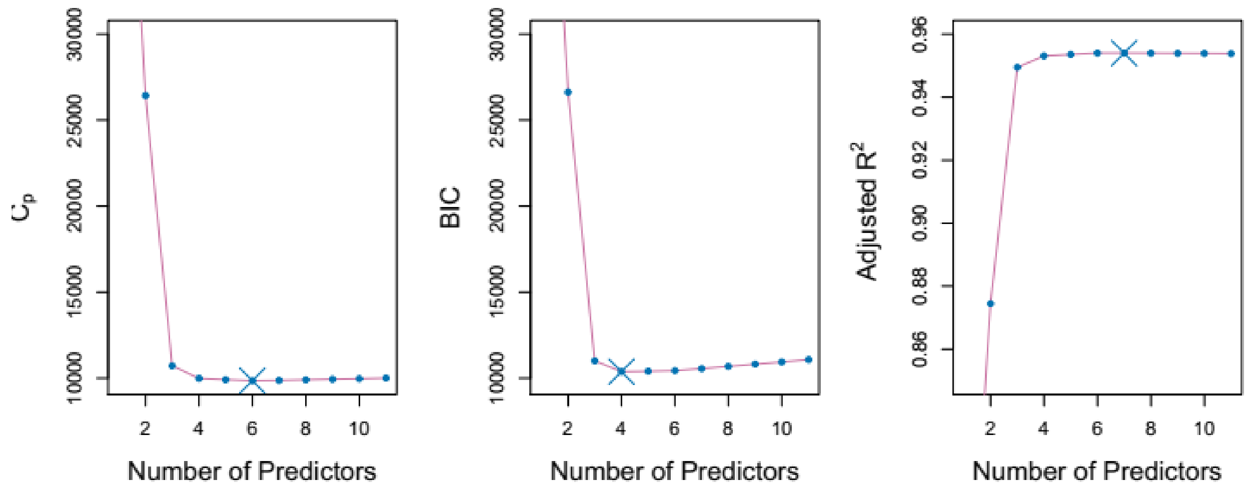
Wider usage

- **AIC** - Takes into account both the goodness of fit and the simplicity of the model. Applicable to a wider range of models, not just linear regression. Relative measure, so it's useful for comparing models but doesn't provide an absolute "good/bad" threshold.
- **BIC** - Due to the heavier penalty, BIC tends to favor simpler models than AIC.



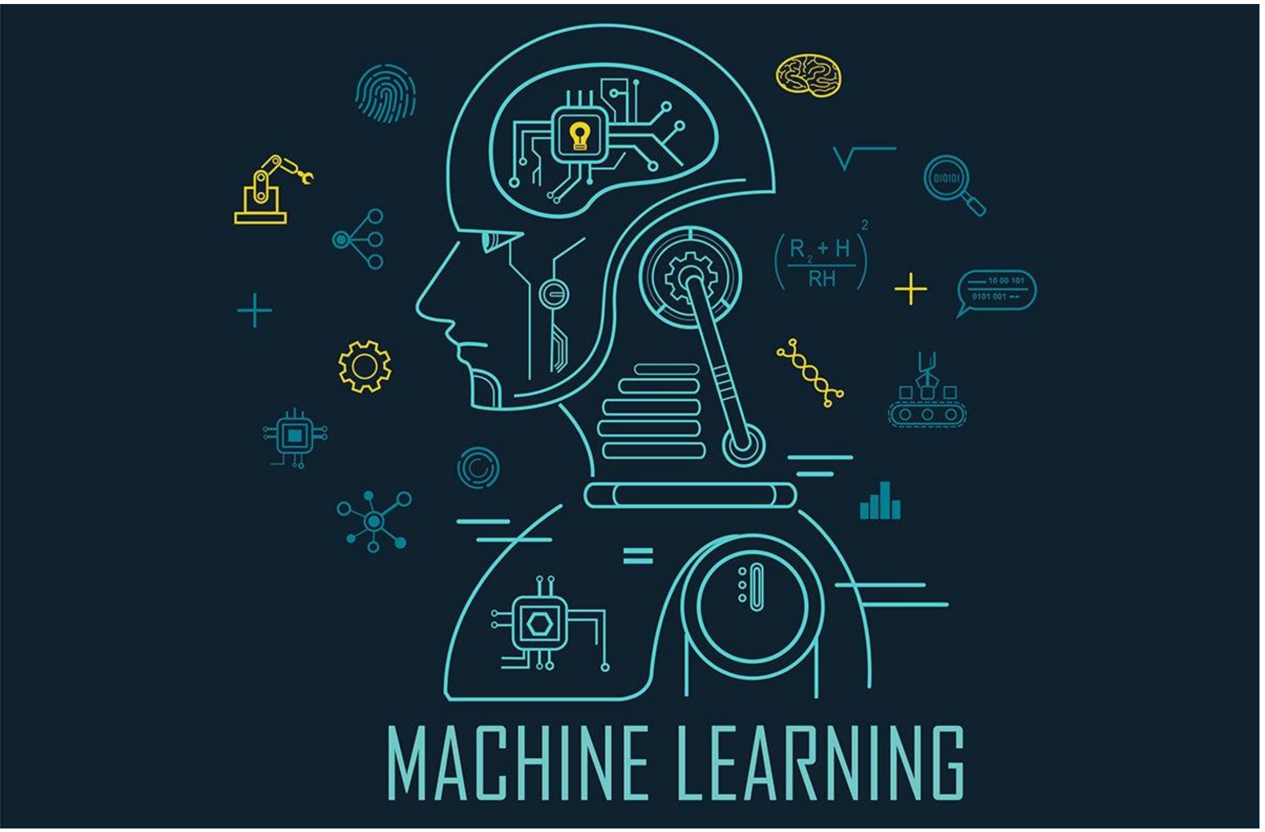
Cross-Validation Error:

- Often, hold-out validation or k-fold cross-validation is used to estimate the model's prediction error on new data.
- Cross-validation provides a more robust measure of a model's predictive performance, especially when you're concerned about overfitting due to adding too many features.



Variance Inflation Factor (VIF):

- Although it's not a performance metric per se, VIF is used to check multicollinearity among features. Multicollinearity can distort the importance of predictors.

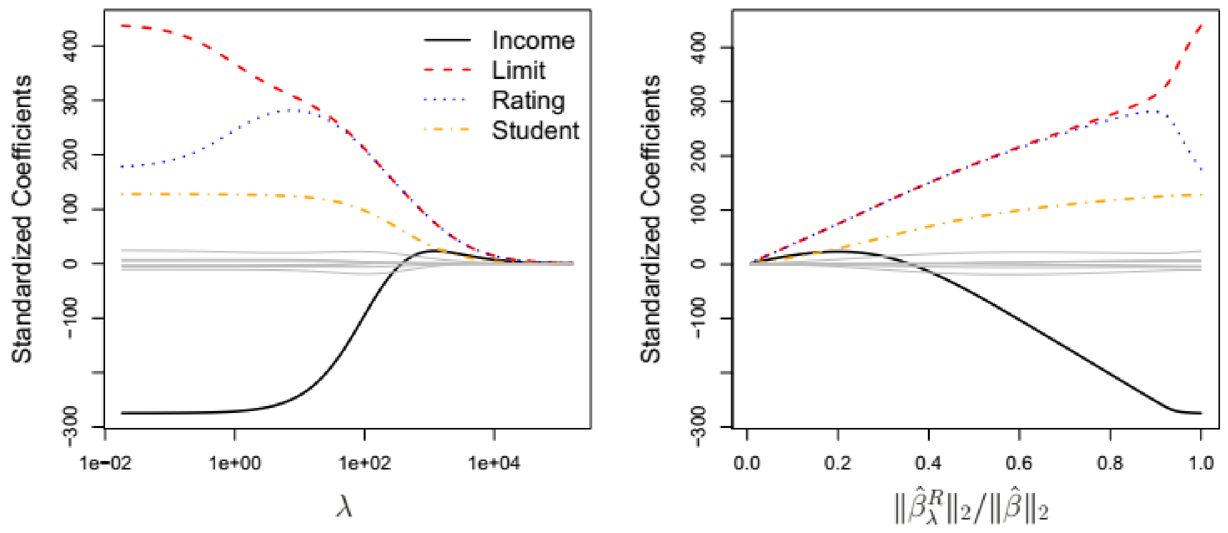


Automated Feature Selection

Shrinkage – constrains or regularizes the coefficient estimates, or equivalently, that shrinks the coefficient estimates towards zero.

- Ridge regression

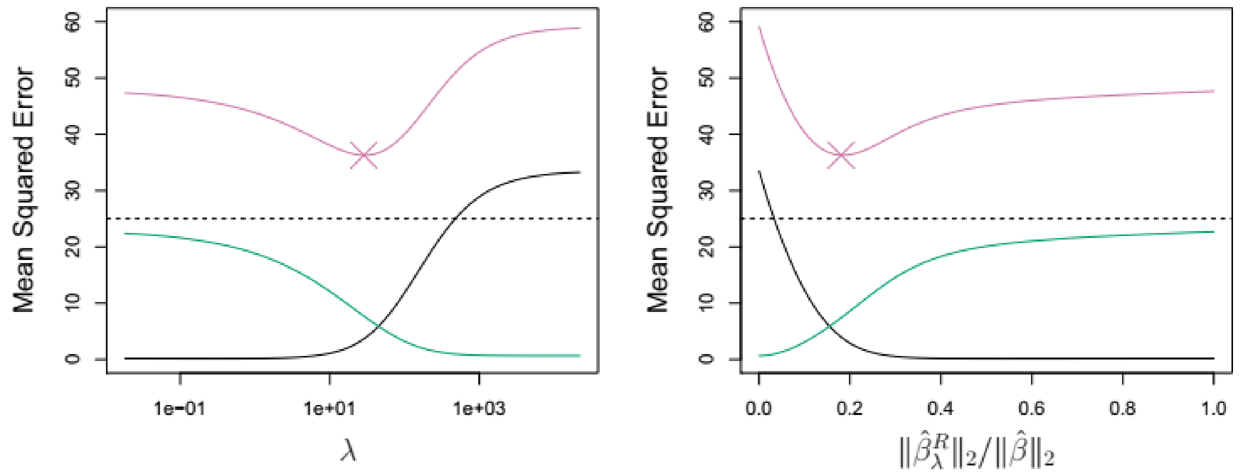
$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2 = \mathrm{RSS} + \lambda\sum_{j=1}^{p}\beta_j^2,$$



Shrinkage – constrains or regularizes the coefficient estimates, or equivalently, that shrinks the coefficient estimates towards zero.

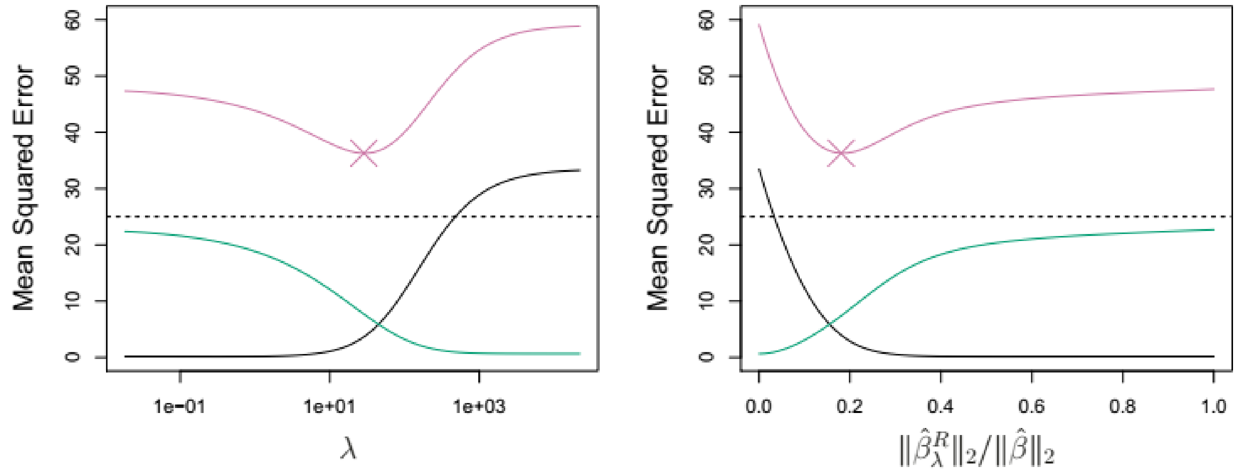
- Ridge regression

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2 = \text{RSS} + \lambda\sum_{j=1}^{p}\beta_j^2,$$



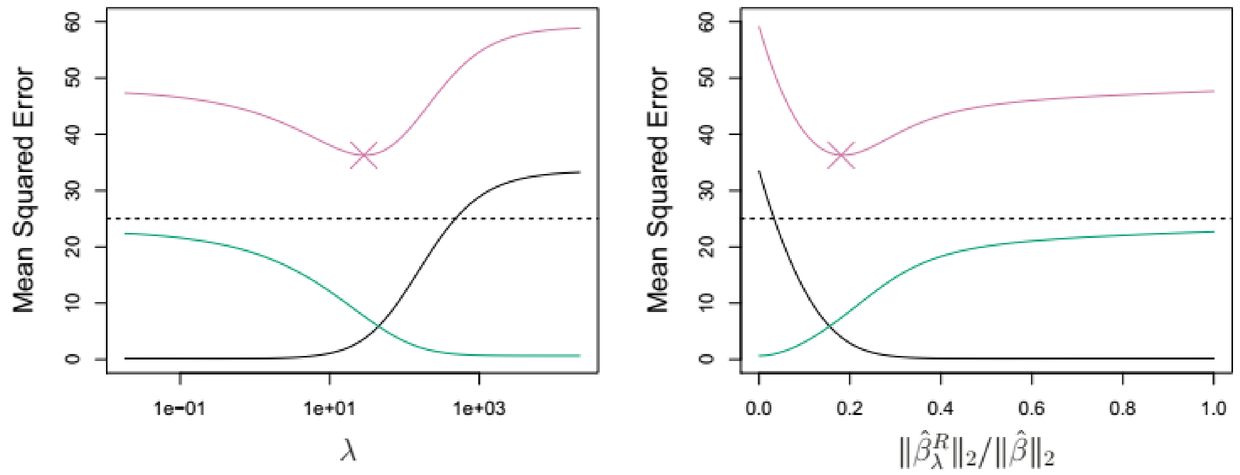
Squared bias (black), variance (green), and test mean squared error (purple)

- Introduces bias by shrinking the coefficients, but this can lead to a significant reduction in variance, resulting in better out-of-sample predictions.
- Helps to reduce the complexity of the model by shrinking the coefficients. This is especially helpful when predictors are correlated (multicollinearity).
- Can help with generalisation.



Squared bias (black), variance (green), and test mean squared error (purple)

- But we don’t actually remove variables / coefficients.

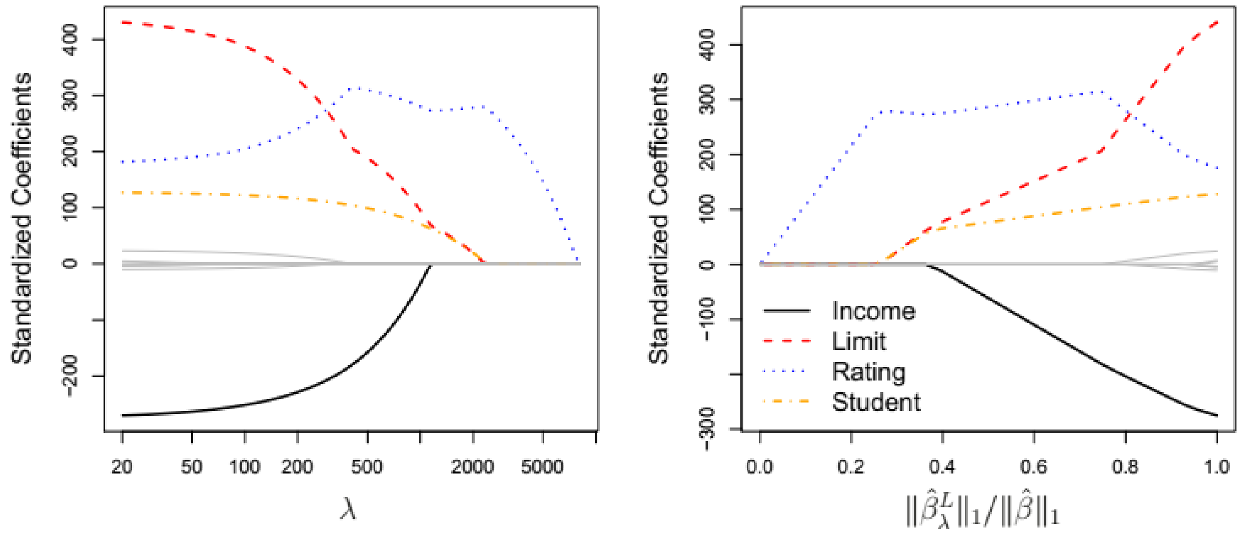


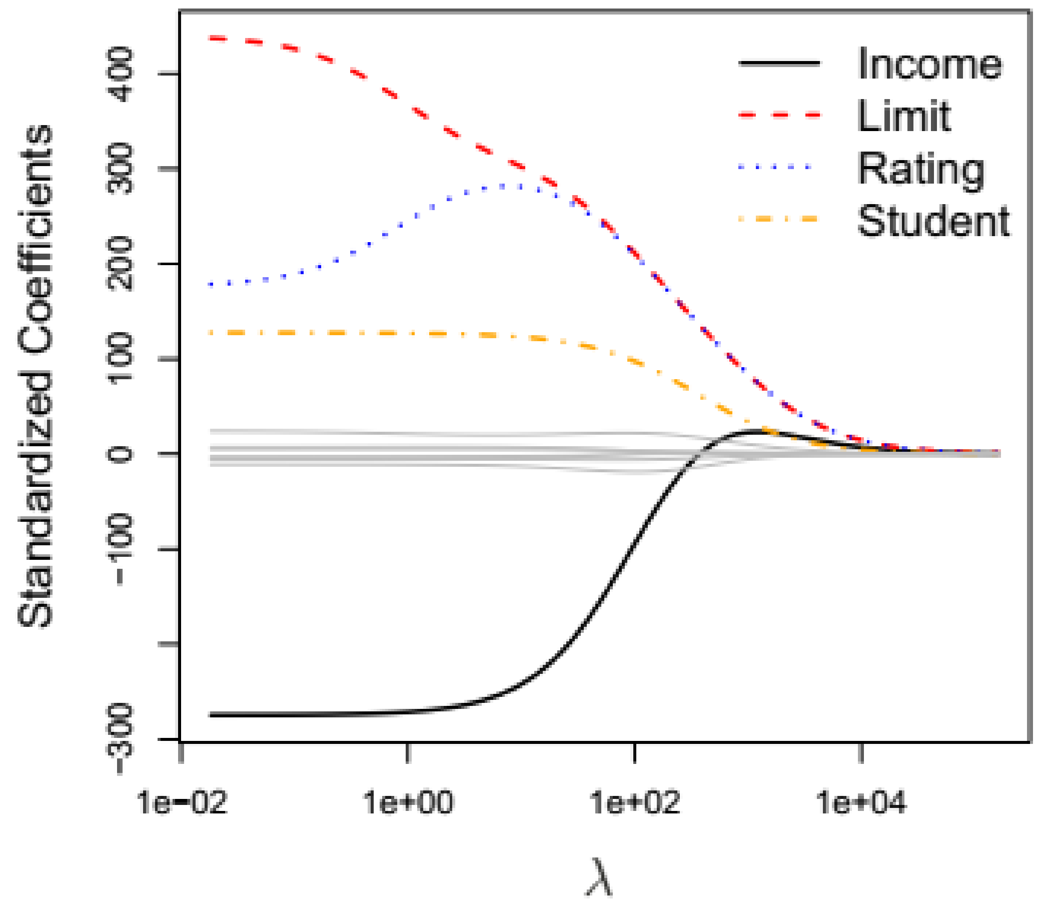
Squared bias (black), variance (green), and test mean squared error (purple)

- The lasso is a relatively recent alternative to ridge regression that overcomes this disadvantage.

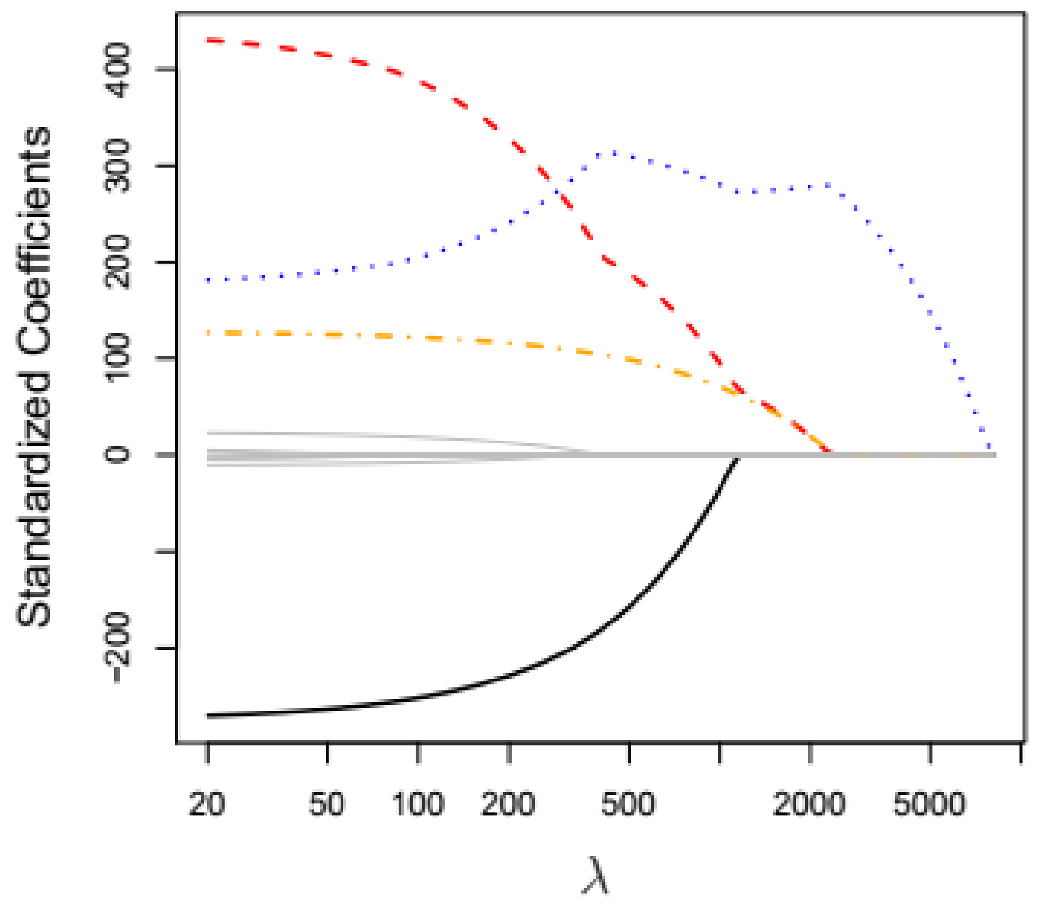
$$\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{p}|\beta_j| \;=\mathrm{RSS}+\lambda\sum_{j=1}^{p}|\beta_j|.$$

- In statistical parlance, the lasso uses an l-1 penalty instead of an l-2 penalty. The l-1 norm of a coefficient vector β is given by $\|\beta\|_1 = \sum|\beta_j|$.



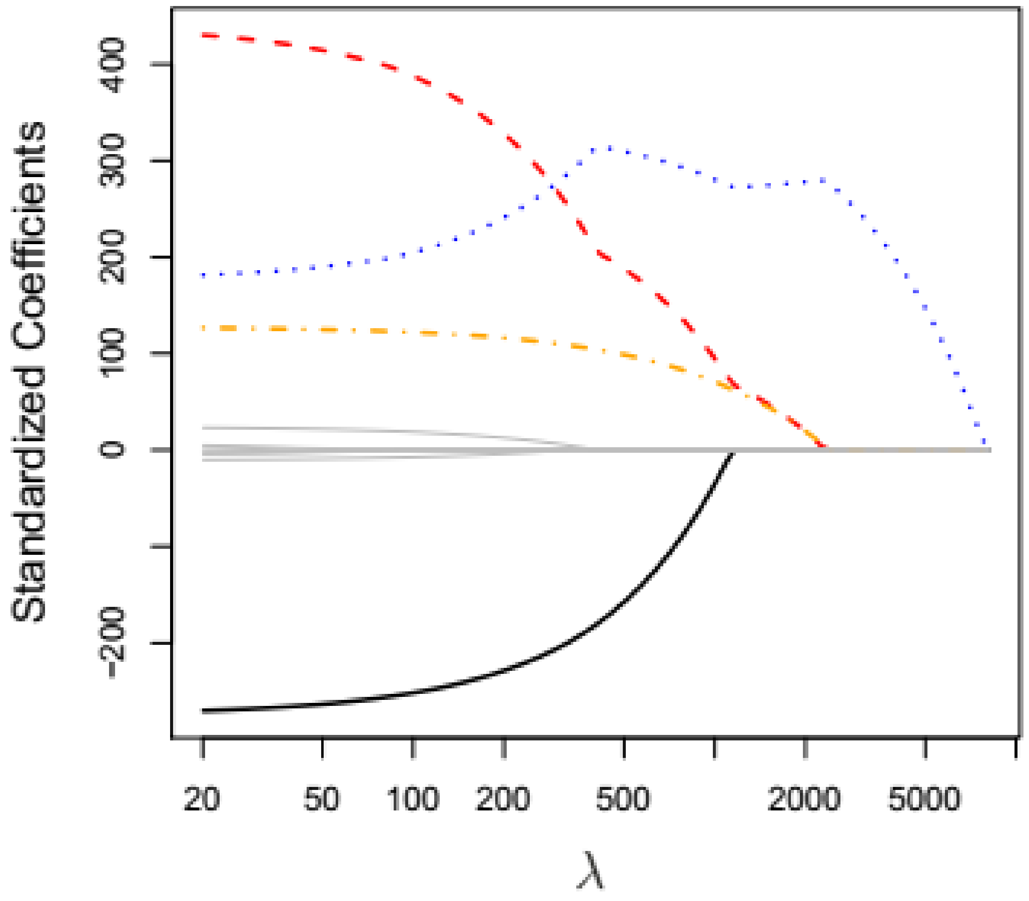


Income
Limit
Rating
Student
Standardized Coefficients
400
300
200
100
0
−100
−300
1e−02
1e+00
1e+02
1e+04
λ



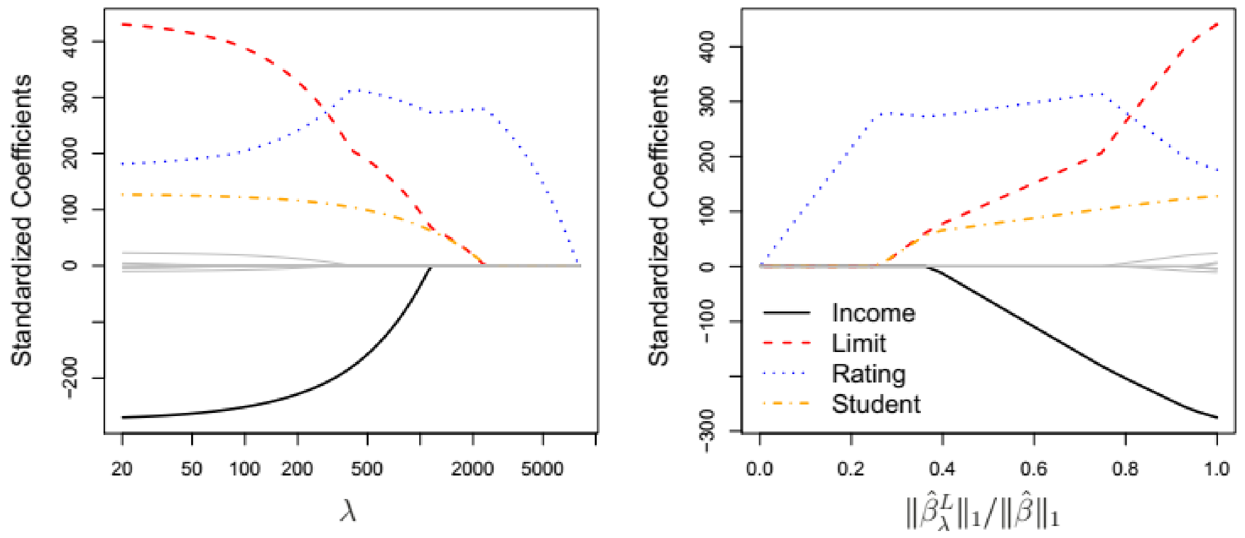
Standardized Coefficients
400
300
200
100
0
−200
20
50
100
200
500
2000
5000
λ

- Check limit and studen



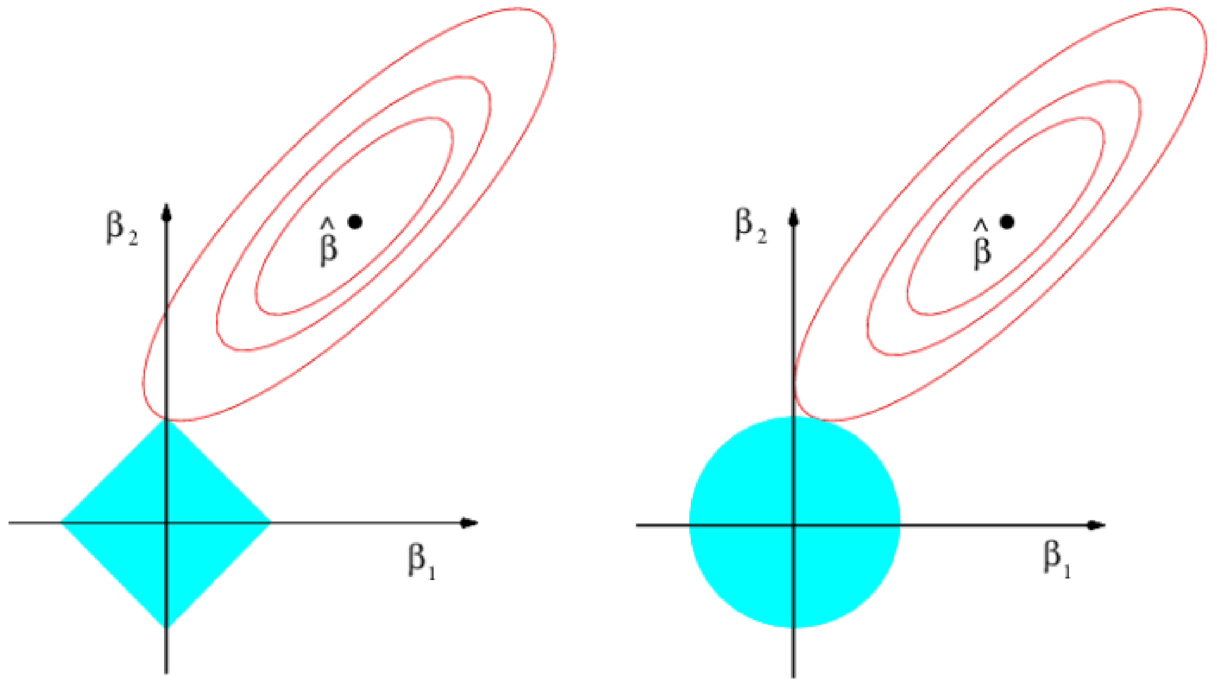
$$\underset{\beta}{\text{minimize}} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \right\} \quad \text{subject to} \quad \sum_{j=1}^{p} |\beta_j| \le s$$

$$\underset{\beta}{\text{minimize}} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \right\} \quad \text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 \le s,$$



$$\underset{\beta}{\text{minimize}} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \right\} \quad \text{subject to} \quad \sum_{j=1}^{p} |\beta_j| \le s$$

$$\underset{\beta}{\text{minimize}} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \right\} \quad \text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 \le s,$$

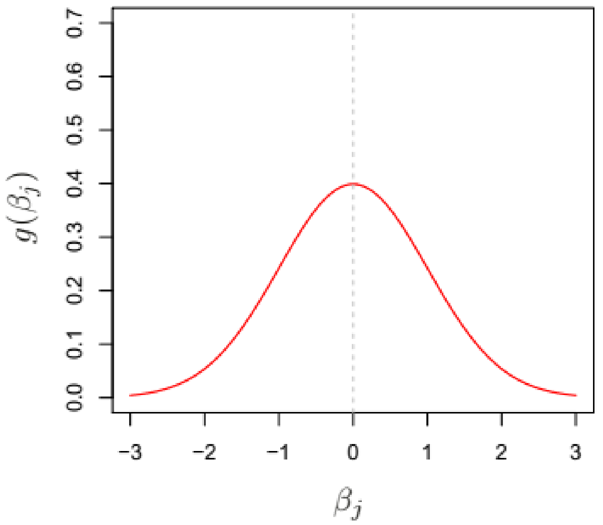


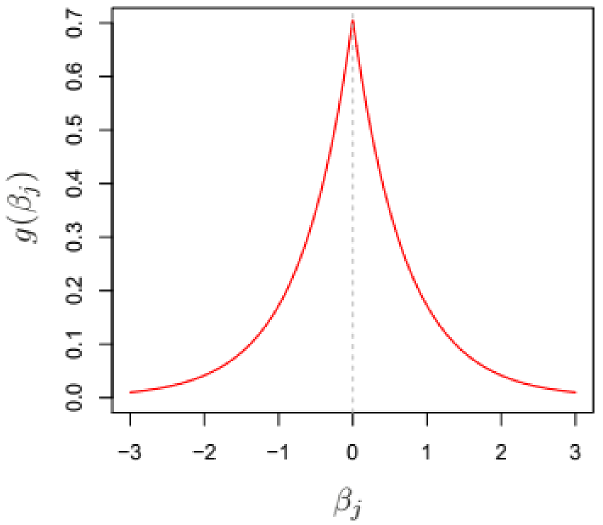
Bayesian priors:

- The lasso expects a priori that many of the coefficients are (exactly) zero

Ridge – Works best when all of our parameters have some usefulness.

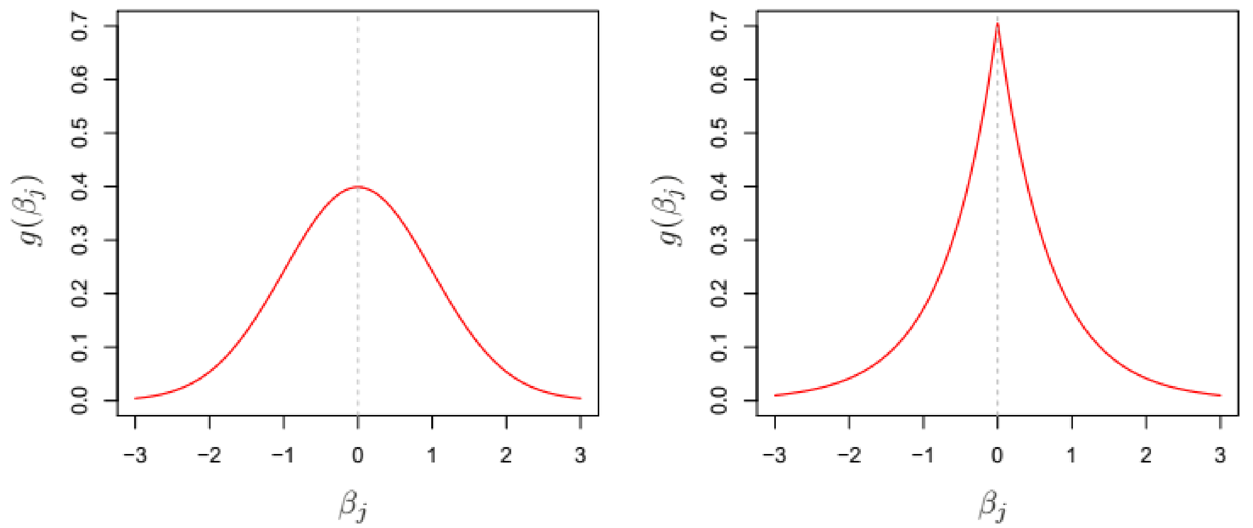
Lasso - Works best when some our parameters should be ruled out. Which can also be for interpretability or inference purposes.





Adjusting coefficients

- Coefficient no longer the effect of x on y
- Relationships less clear, biased lower.

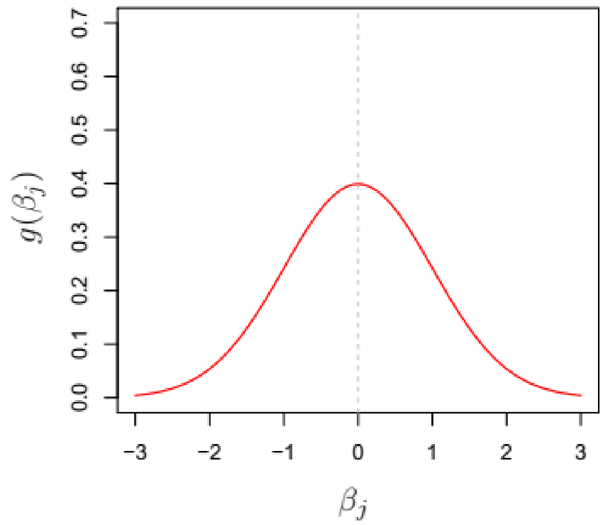


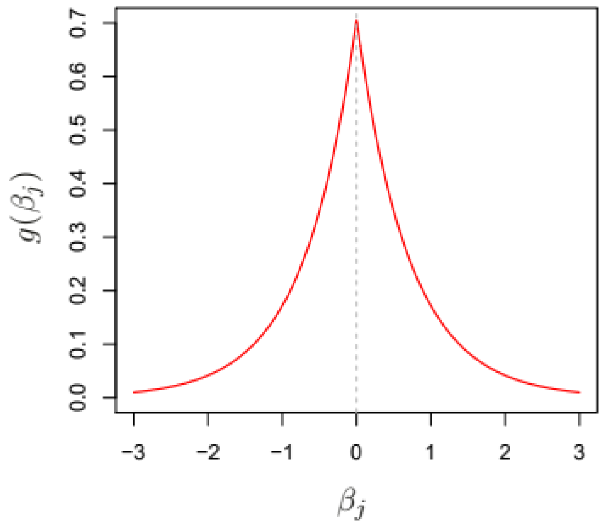
Common workflow:

- Use ridge/lasso to select or stabilize predictors
- Refit **OLS on the selected variables**
- Report standard errors, confidence intervals, p-values

Problem:

Anti conservative - "Because variable selection was performed on the same dataset used for estimation, the reported p-values and confidence intervals are conditional on the selected model and may understate true uncertainty."



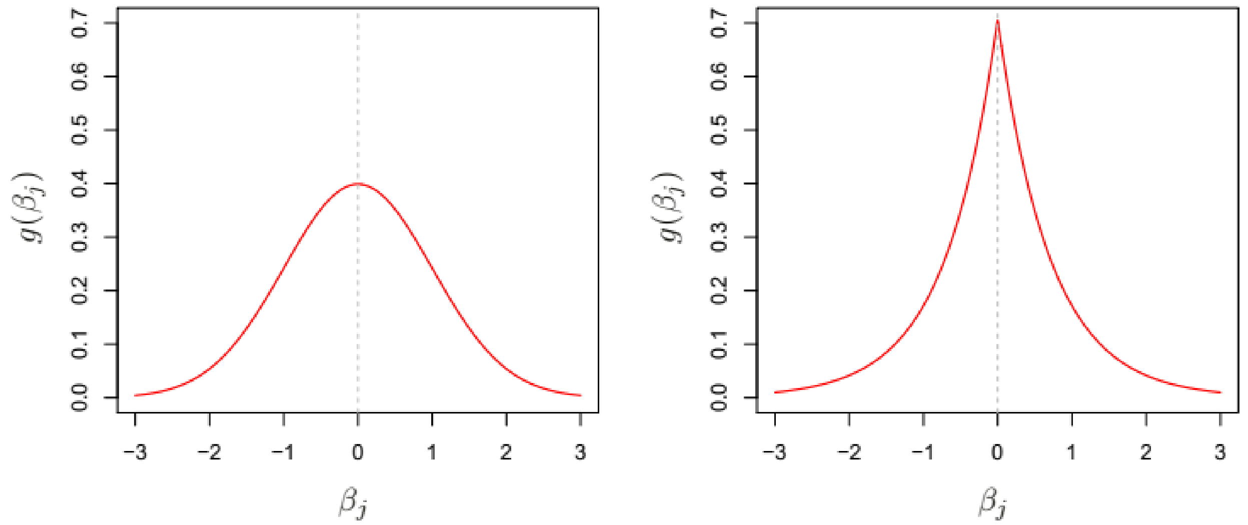


Common workflow:

- Use ridge/lasso to select or stabilize predictors
- Refit **OLS on the selected variables**
- Report standard errors, confidence intervals, p-values

Problem:

LEAKY if done on full dataset!



Common workflow:

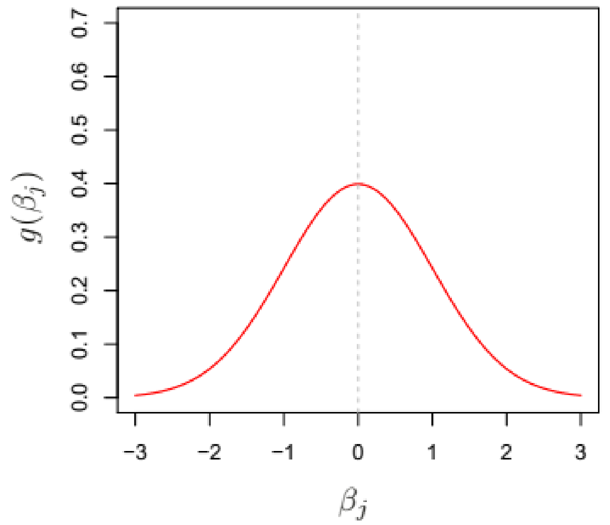
- Use ridge/lasso to select or stabilize predictors
- Apply **ML model (trees, neural network etc).**

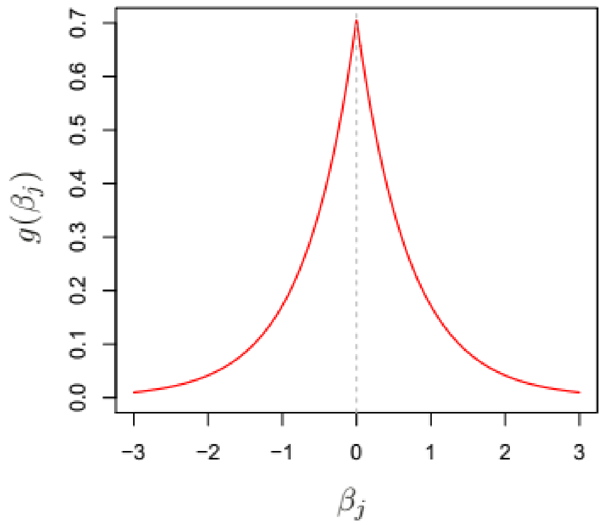
Problem:

- Variable selection done in simple linear model (coefficients not useful in a linear model)
- Advantage of ML models is they can find non-linearities that we may be discarding.
- Trees, NN, have other ways to regularise internally.

Maybe acceptable:

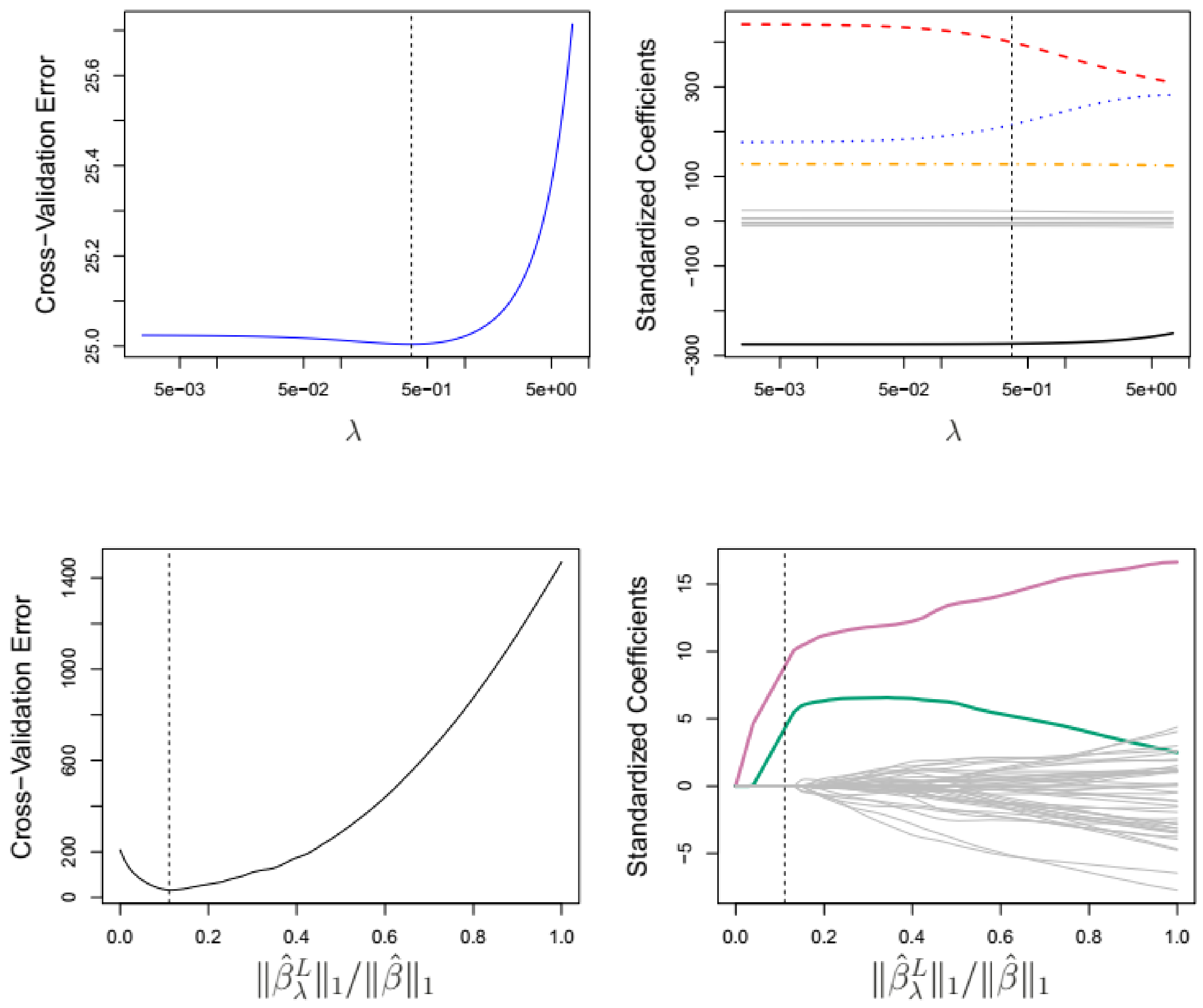
- Transforming thousands of features to hundreds





How do we set our parameters?

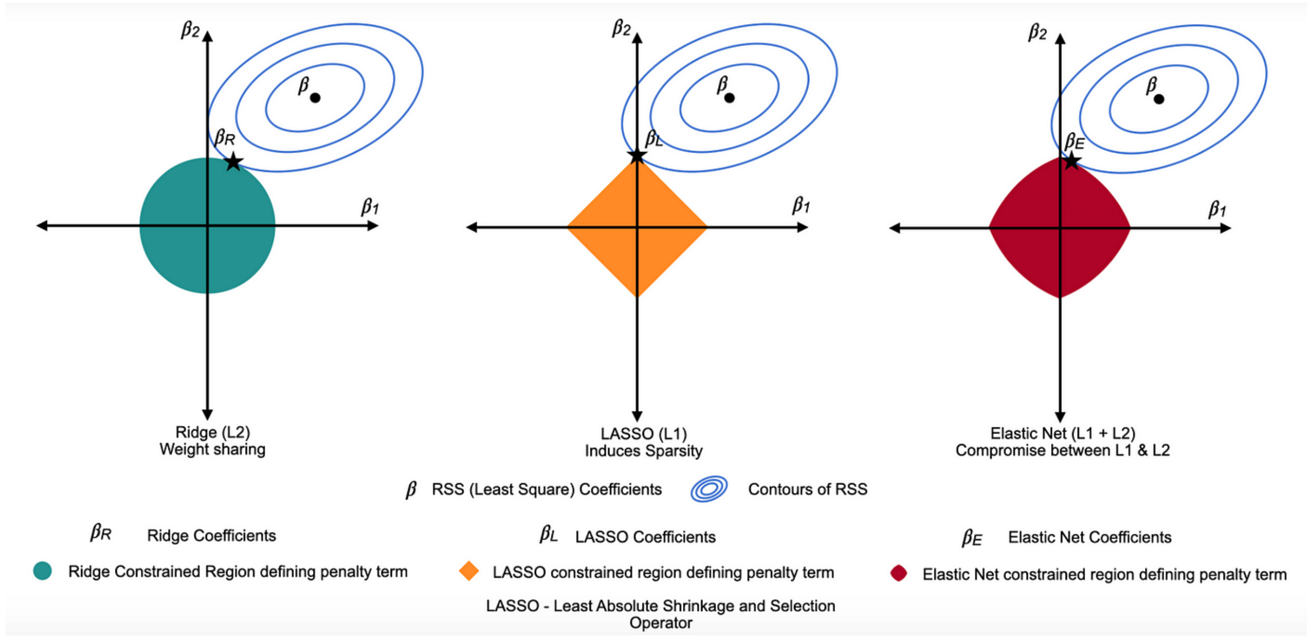
- Hyperparameter tuning with CV



What if we want both?

Elastic Net Regression – Weighted average of penalty terms.

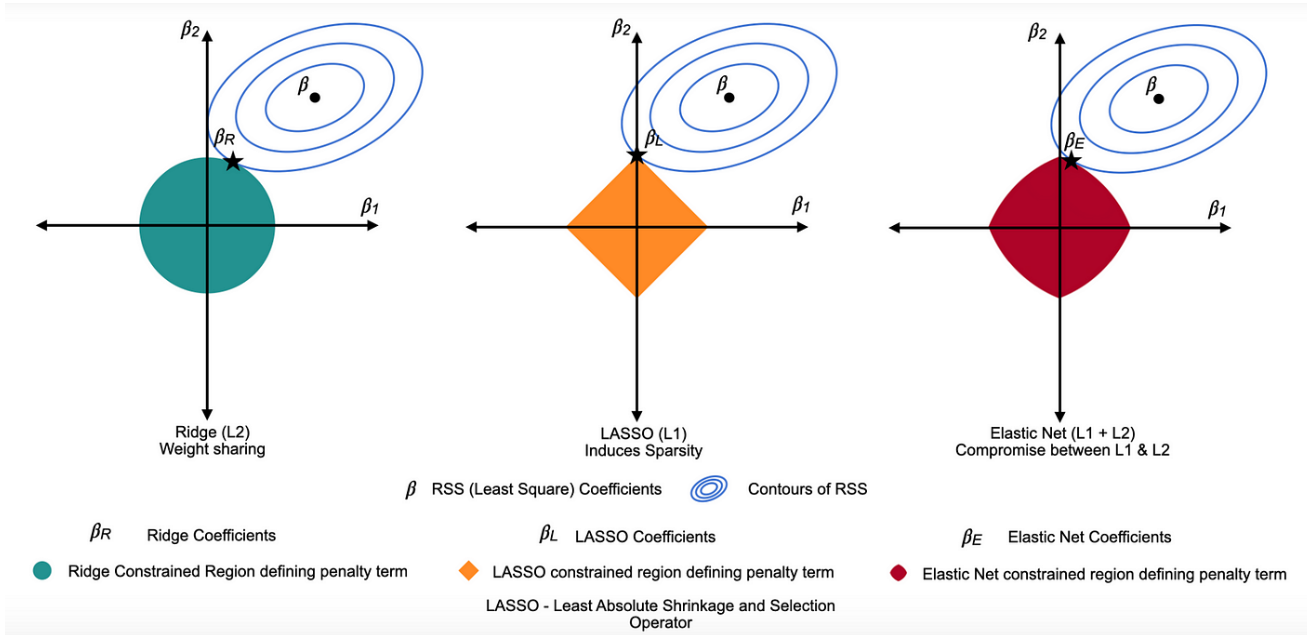
Handles Multicollinearity better than Lasso alone, because the Ridge component helps stabilize coefficient estimates among correlated features.


β_2
β_1
β
β_R
β_L
β_E
Ridge (L2)
Weight sharing
LASSO (L1)
Induces Sparsity
Elastic Net (L1 + L2)
Compromise between L1 & L2
β RSS (Least Square) Coefficients
Contours of RSS
β_R Ridge Coefficients
β_L LASSO Coefficients
β_E Elastic Net Coefficients
● Ridge Constrained Region defining penalty term
◆ LASSO constrained region defining penalty term
◆ Elastic Net constrained region defining penalty term
LASSO - Least Absolute Shrinkage and Selection Operator

Overall Regularization Strength (α) - Scales how strongly you penalize large coefficients.

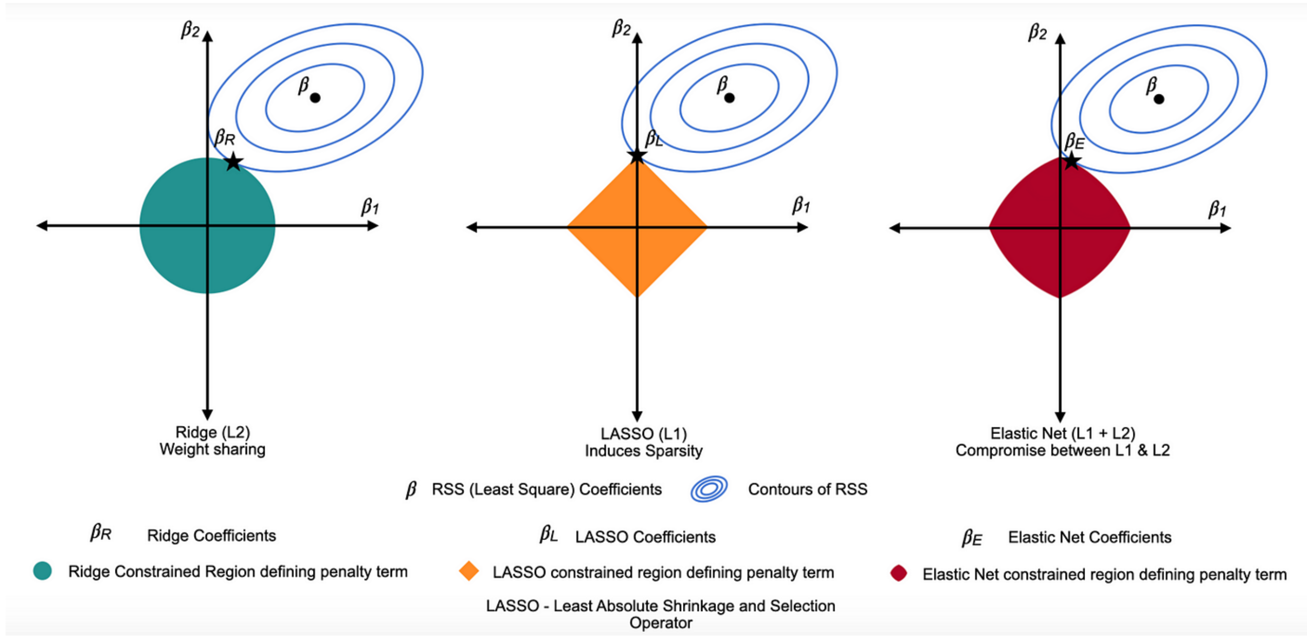
Ratio ($l1\_ratio$) - Determines the relative proportion of L1 vs. L2. Ranges from 0 to 1:

- $l1\_ratio = 0 \Rightarrow$ Pure Ridge
- $l1\_ratio = 1 \Rightarrow$ Pure Lasso


β_2
β_1
β
β_R
β_L
β_E
Ridge (L2)
Weight sharing
LASSO (L1)
Induces Sparsity
Elastic Net (L1 + L2)
Compromise between L1 & L2
β RSS (Least Square) Coefficients
Contours of RSS
β_R Ridge Coefficients
β_L LASSO Coefficients
β_E Elastic Net Coefficients
Ridge Constrained Region defining penalty term
LASSO constrained region defining penalty term
Elastic Net constrained region defining penalty term
LASSO - Least Absolute Shrinkage and Selection Operator

Lasso Least Angle Regression (LARS) - LARS is a specialized algorithm for fitting the Lasso model.

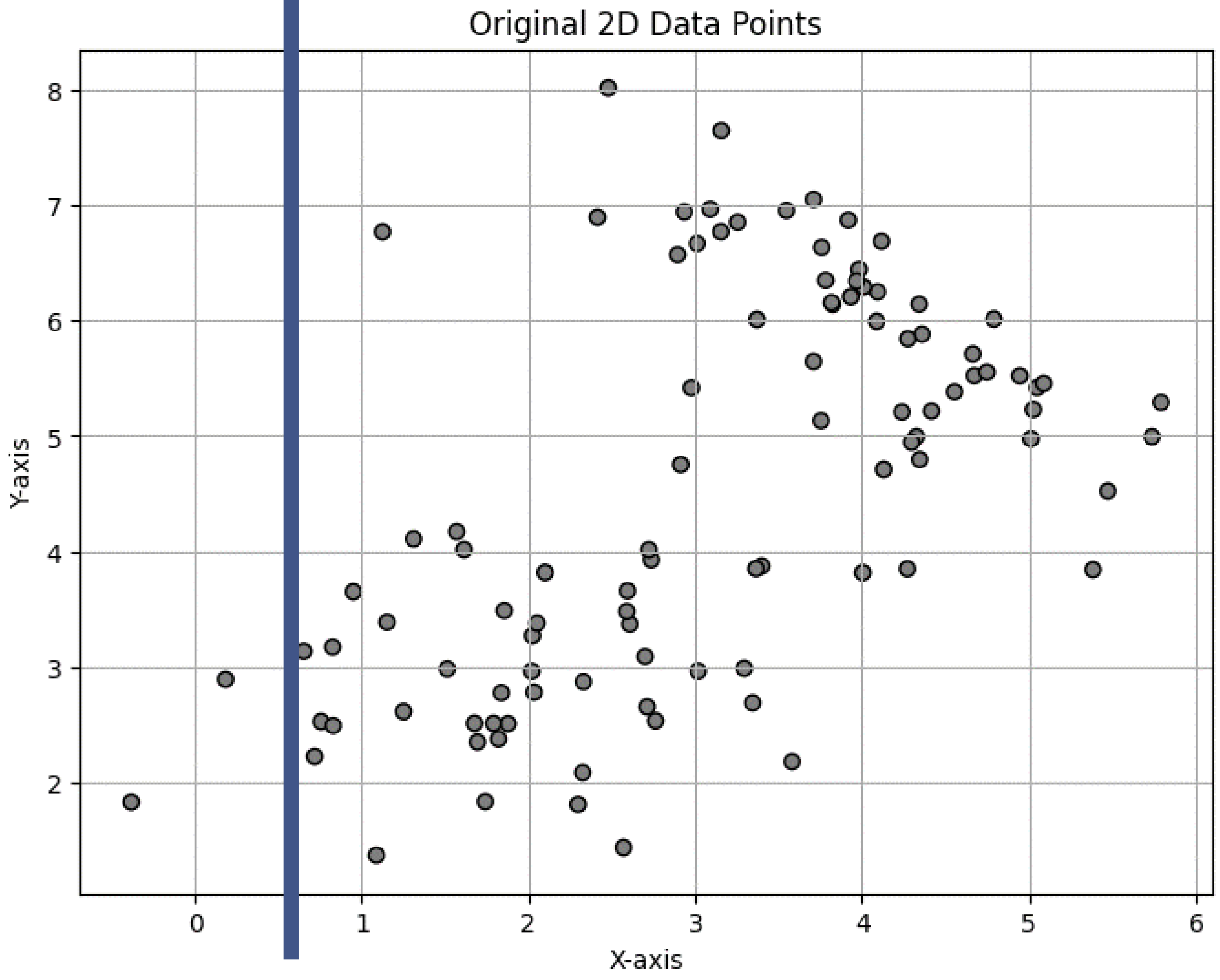
- It is particularly efficient when the number of predictors p is much larger than the number of samples n (high-dimensional data).
- It has a step that starts by adding variables rather than reducing, so can lead to different solutions. We might expect these should be better in high dimensional better.
- Performs, differently with highly correlated predictors, more likely to include both, at least to start.

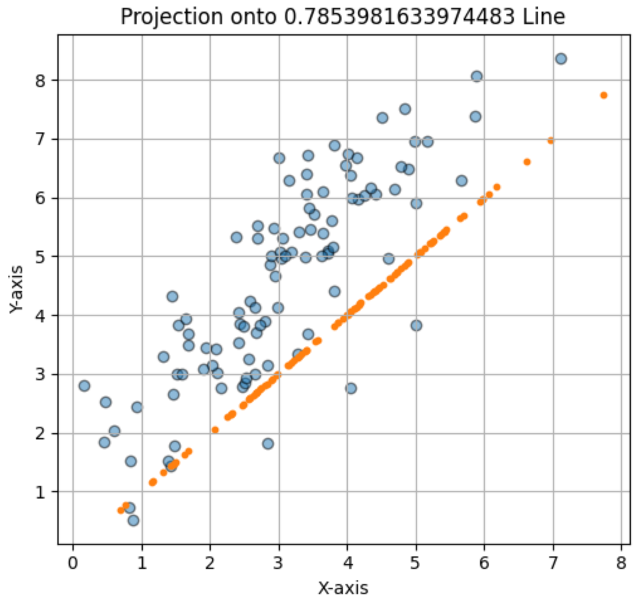
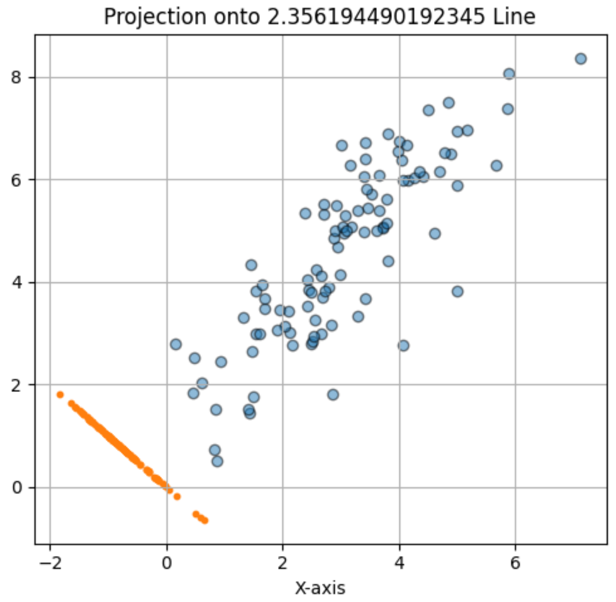
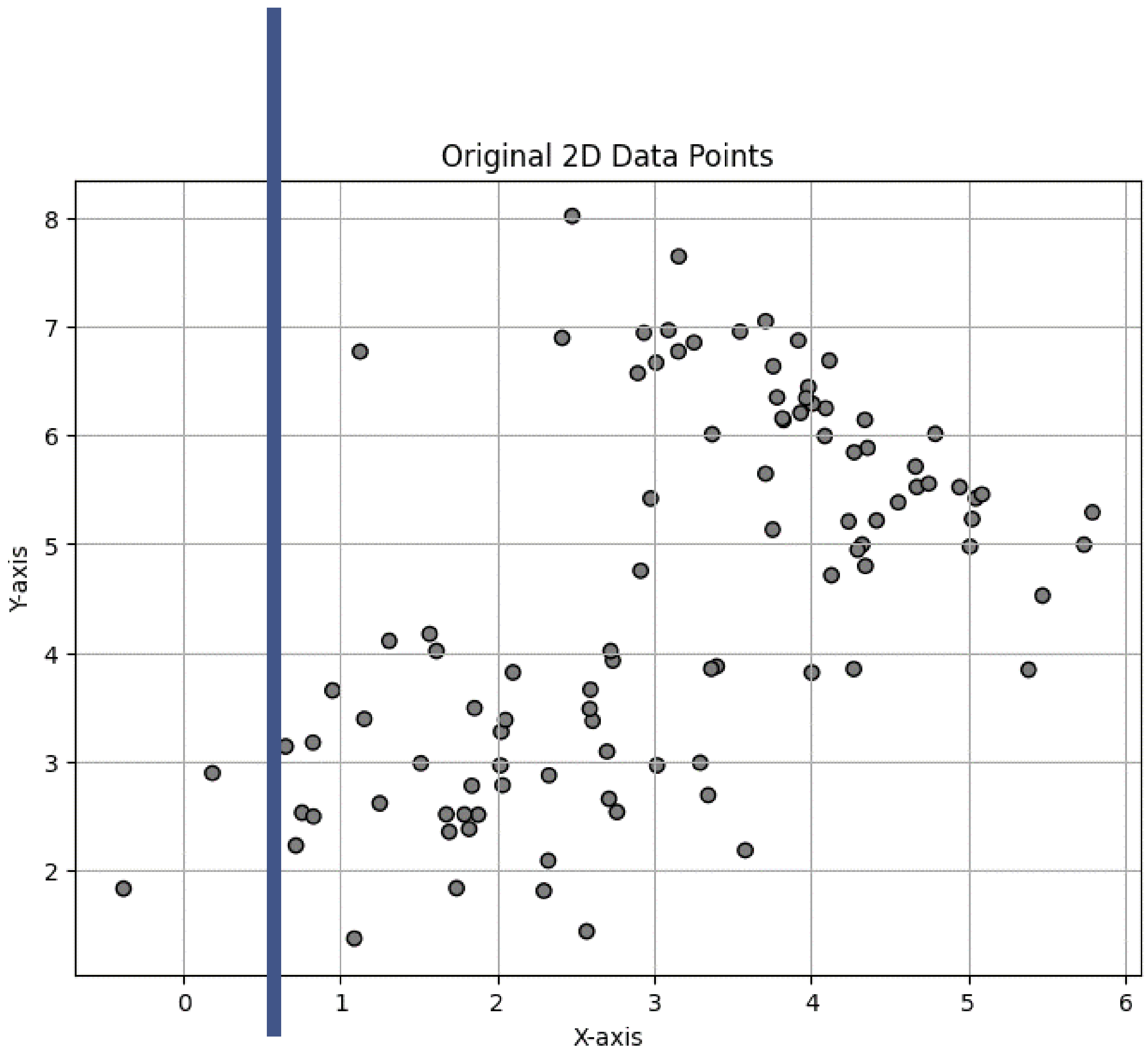


β_2
β
β_R
β_1
Ridge (L2)
Weight sharing
β_L
LASSO (L1)
Induces Sparsity
β_E
Elastic Net (L1 + L2)
Compromise between L1 & L2
β RSS (Least Square) Coefficients
Contours of RSS
β_R Ridge Coefficients
β_L LASSO Coefficients
β_E Elastic Net Coefficients
● Ridge Constrained Region defining penalty term
◆ LASSO constrained region defining penalty term
◆ Elastic Net constrained region defining penalty term
LASSO - Least Absolute Shrinkage and Selection Operator

Unsupervised learning - **Dimensionality reduction.**

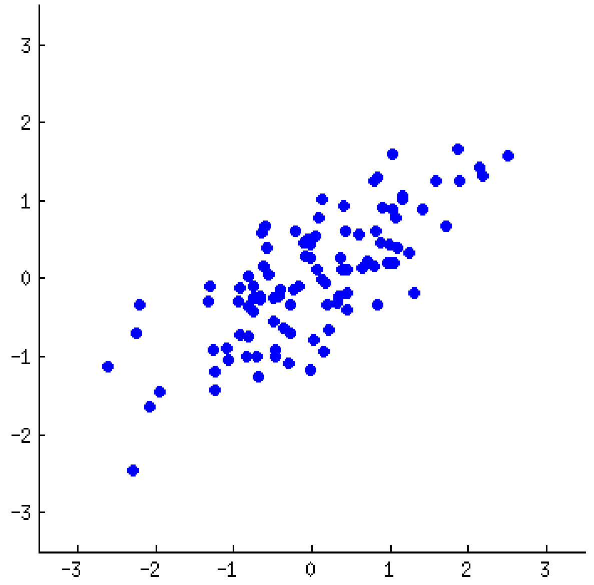
Principle component regression

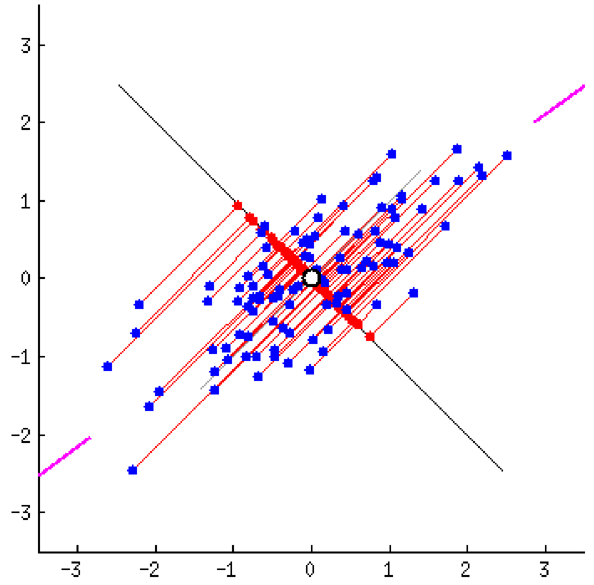



Projection onto 0.7853981633974483 Line
X-axis
Y-axis

Projection onto 2.356194490192345 Line
X-axis

Original 2D Data Points
X-axis
Y-axis

- Linear combinations of existing variables – multiple variables become a single variable

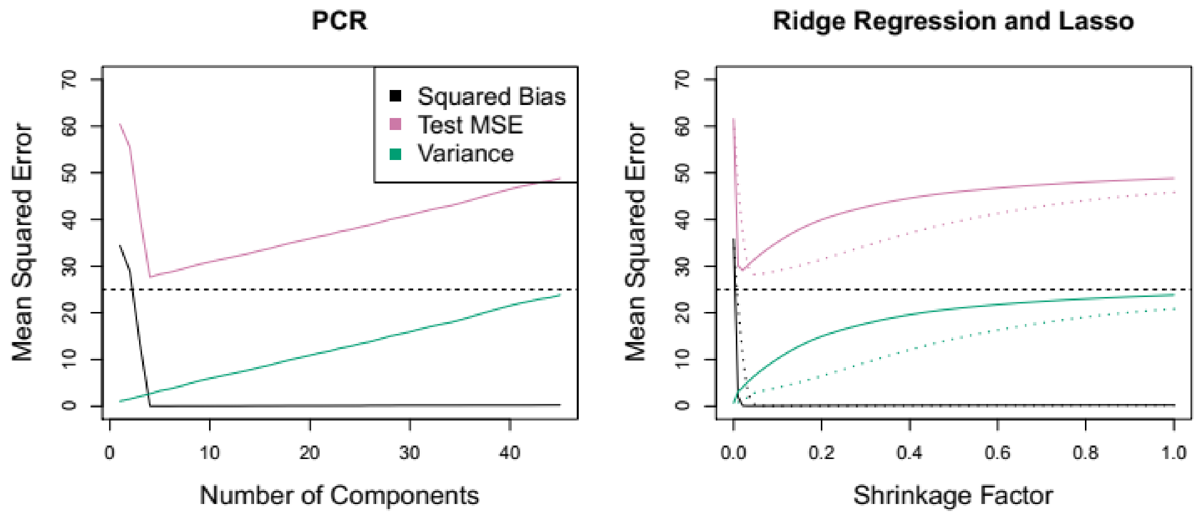
$$Z_2 = 0.544 \times (\mathtt{pop} - \overline{\mathtt{pop}}) - 0.839 \times (\mathtt{ad} - \overline{\mathtt{ad}}).$$



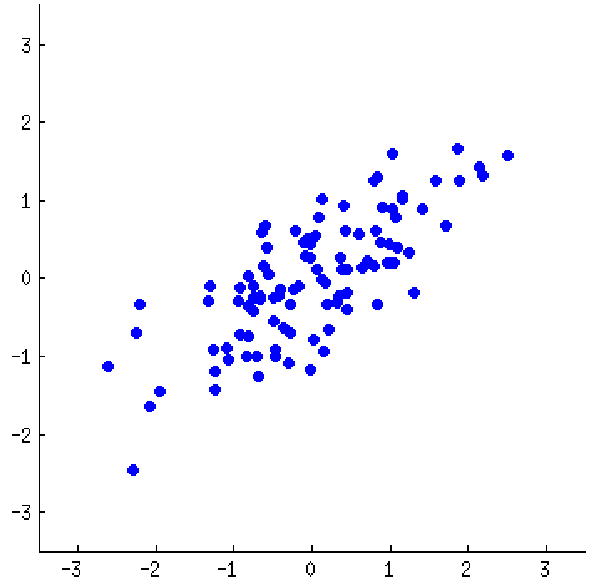


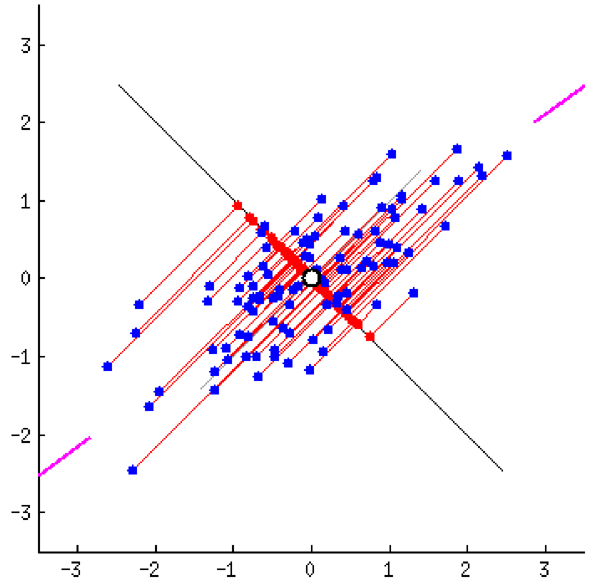
Principle component regression

- Select first m components
- Use CV to pick m
- Standardise first



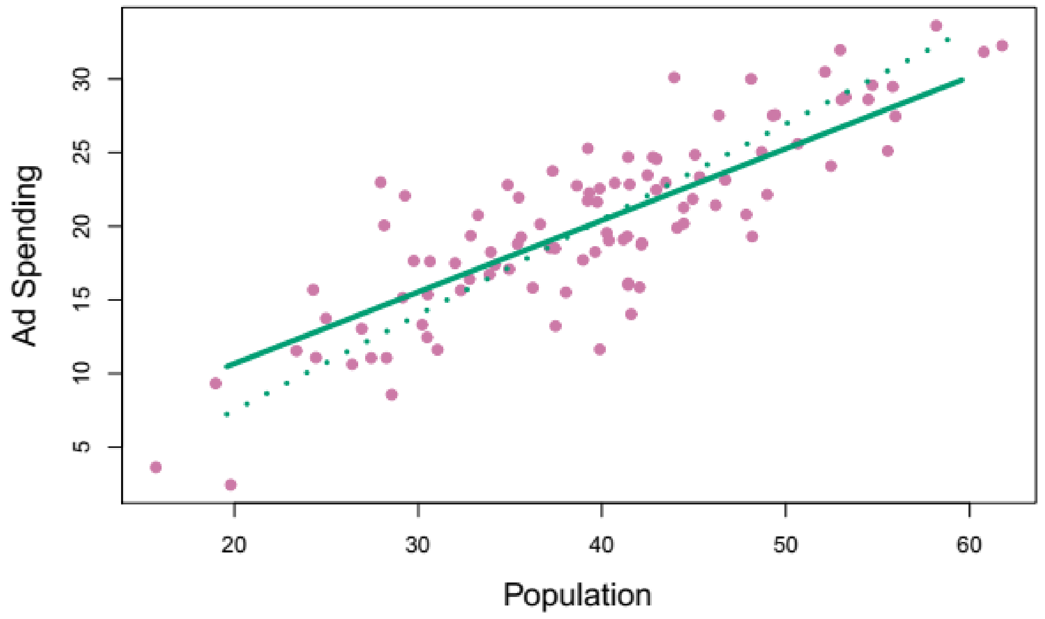
- Identify linear combinations, or directions, that best represent the predictors
- These directions are identified in an unsupervised way, since the response Y is not used to help determine the principal component directions. That is, the response does not supervise the identification of the principal components.
- No guarantee these directions make sense for prediction.
- Lose interpretability





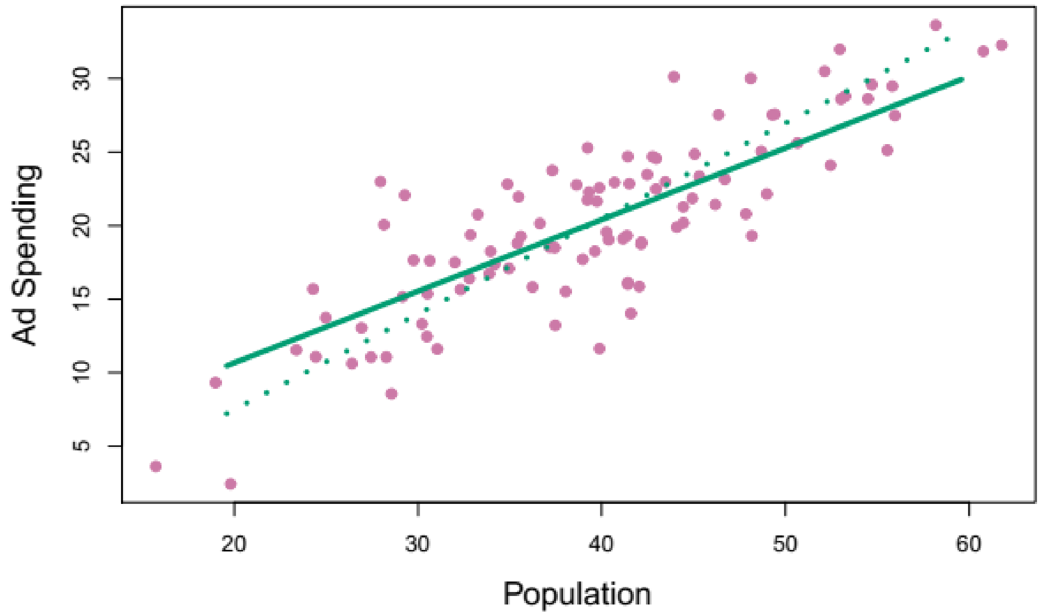
Partial least squares (PLS)

- PLS identifies these new features in a supervised way—that is, it makes use of the response Y in order to identify new features that not only approximate the old features well, but also that are related to the response.
- PLS places the highest weight on the variables that are most strongly related to the response.



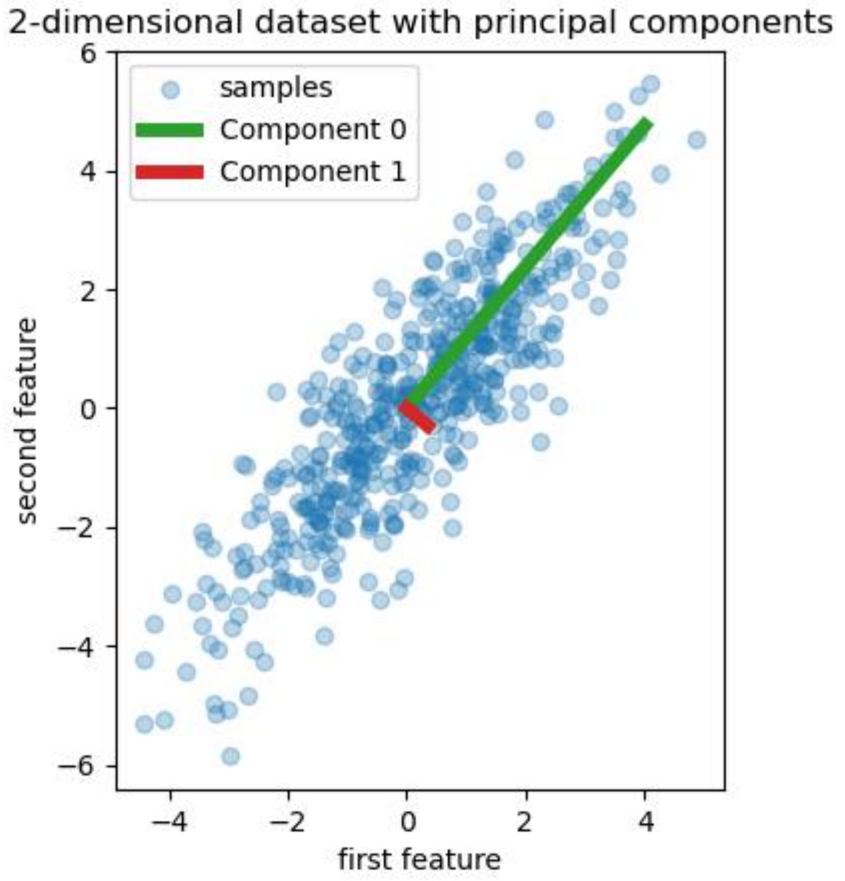
PLS direction (solid line) and first PCR direction (dotted line) are shown.
Trying to represent more population.

- PLS is popular in the field of chemometrics, where many variables arise from digitized spectrometry signals. In practice it often performs no better than ridge regression or PCR.
- While the supervised dimension reduction of PLS can reduce bias, it also has the potential to increase variance, so that the overall benefit of PLS relative to PCR is a wash.

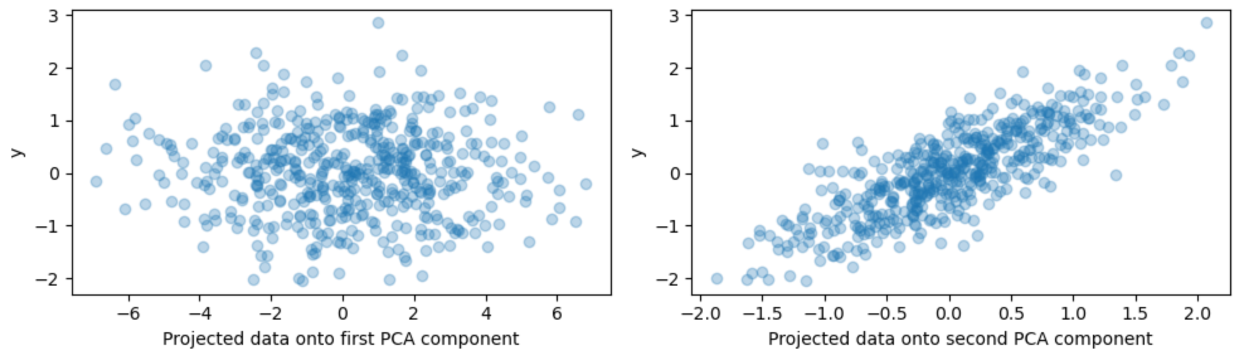


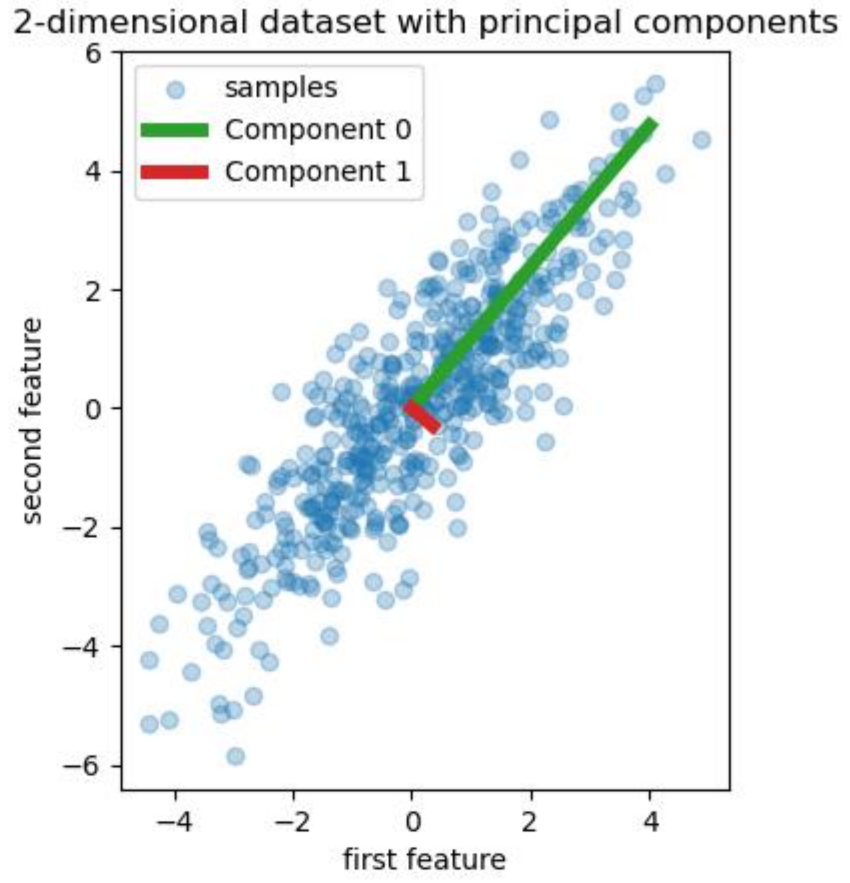
PLS direction (solid line) and first PCR direction (dotted line) are shown.
Trying to represent more population.

- Example from scikitlearn comparing PCR and PLS.

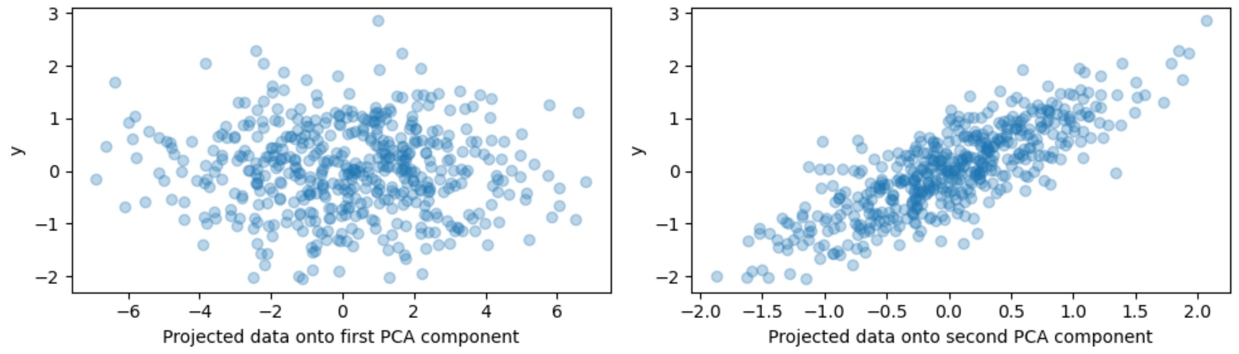


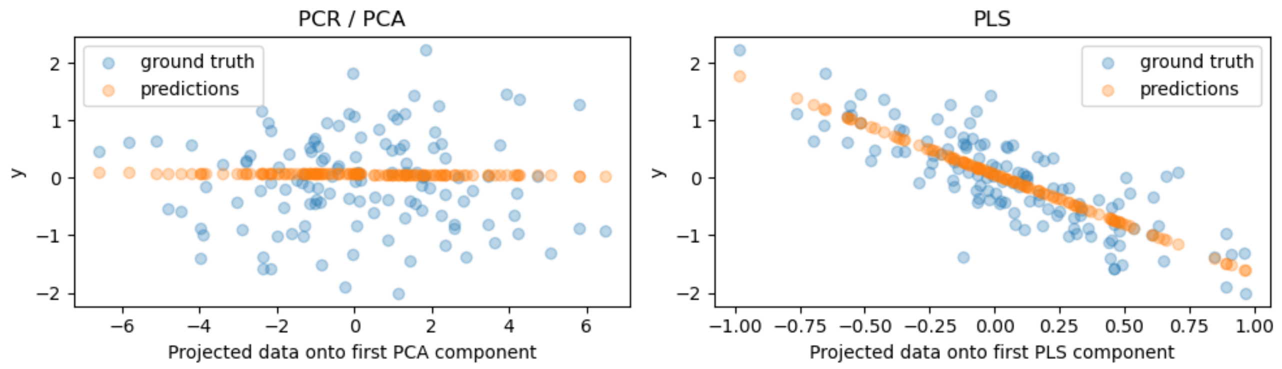
nttps://scikit-learn.org/1.5/auto_examples/cross_decomposition/plot_pcr_vs_pls.html#sphx-glr-auto-examples-cross-decomposition-plot-pcr-vs-pls-py





nttps://scikit-learn.org/1.5/auto_examples/cross_decomposition/plot_pcr_vs_pls.html#sphx-glr-auto-examples-cross-decomposition-plot-pcr-vs-pls-py

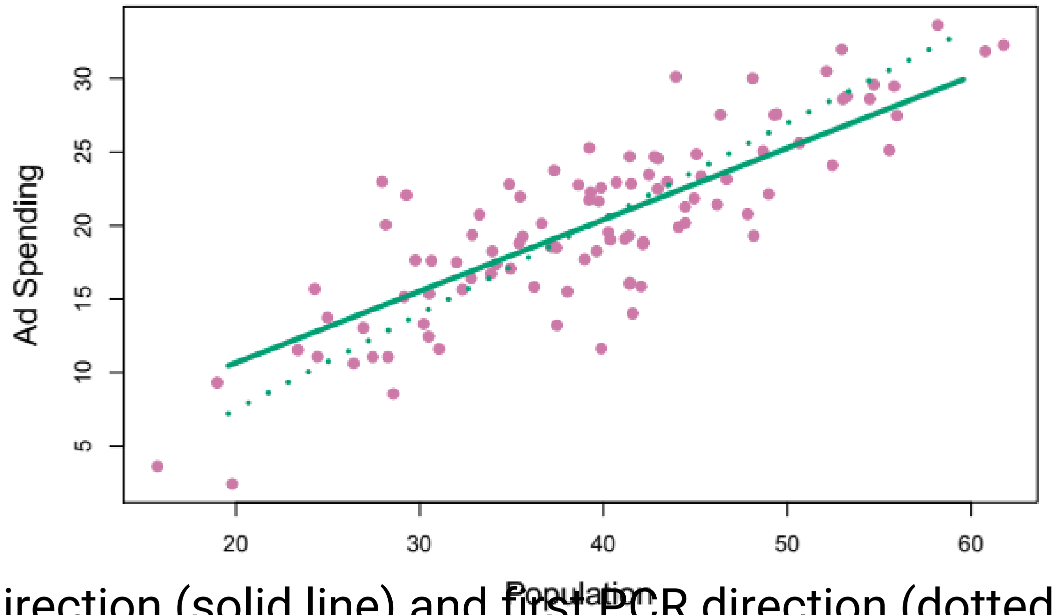




```
PCR r-squared -0.026
PLS r-squared 0.658
```

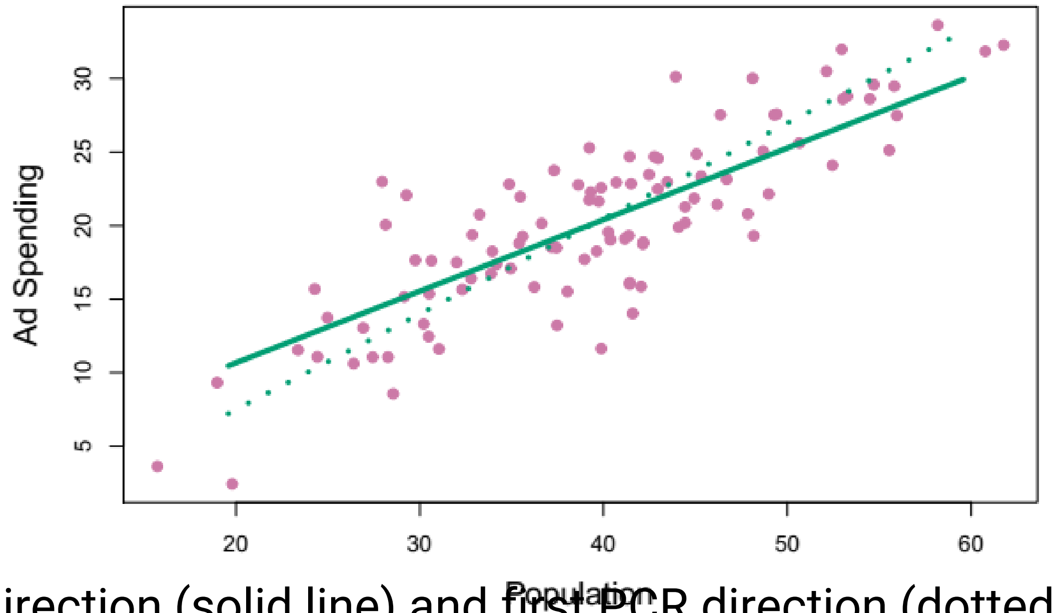
https://scikit-learn.org/1.5/auto_examples/cross_decomposition/plot_pcr_vs_pls.html#sphx-glr-auto-examples-cross-decomposition-plot-pcr-vs-pls-py

We can still do **inference**. Each component is a linear combination of features and the model a linear combination of components.



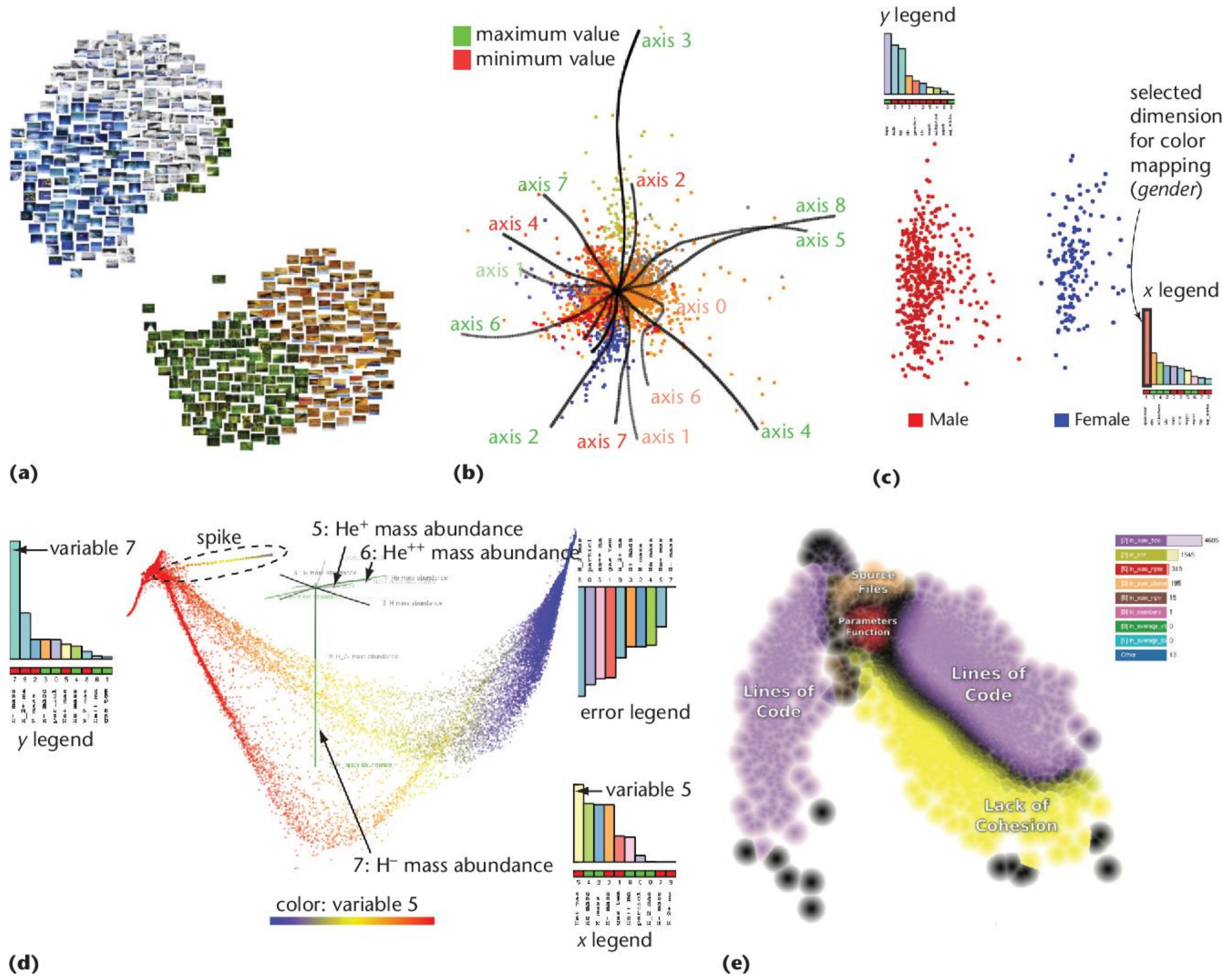
PLS direction (solid line) and first PCR direction (dotted line) are shown.
Trying to represent more population.

- These techniques help us deal with HIGH DIMENISONAL DATA
- Imagine a marketing analyst interested in understanding people's online shopping patterns could treat as features all of the search terms entered by users of a search engine. This is sometimes known as the "bag-of-words" model. For a given user, each of the p search terms is scored present (0) or absent (1), creating a large binary feature vector. Then n ≈ 1,000 and p is much larger.

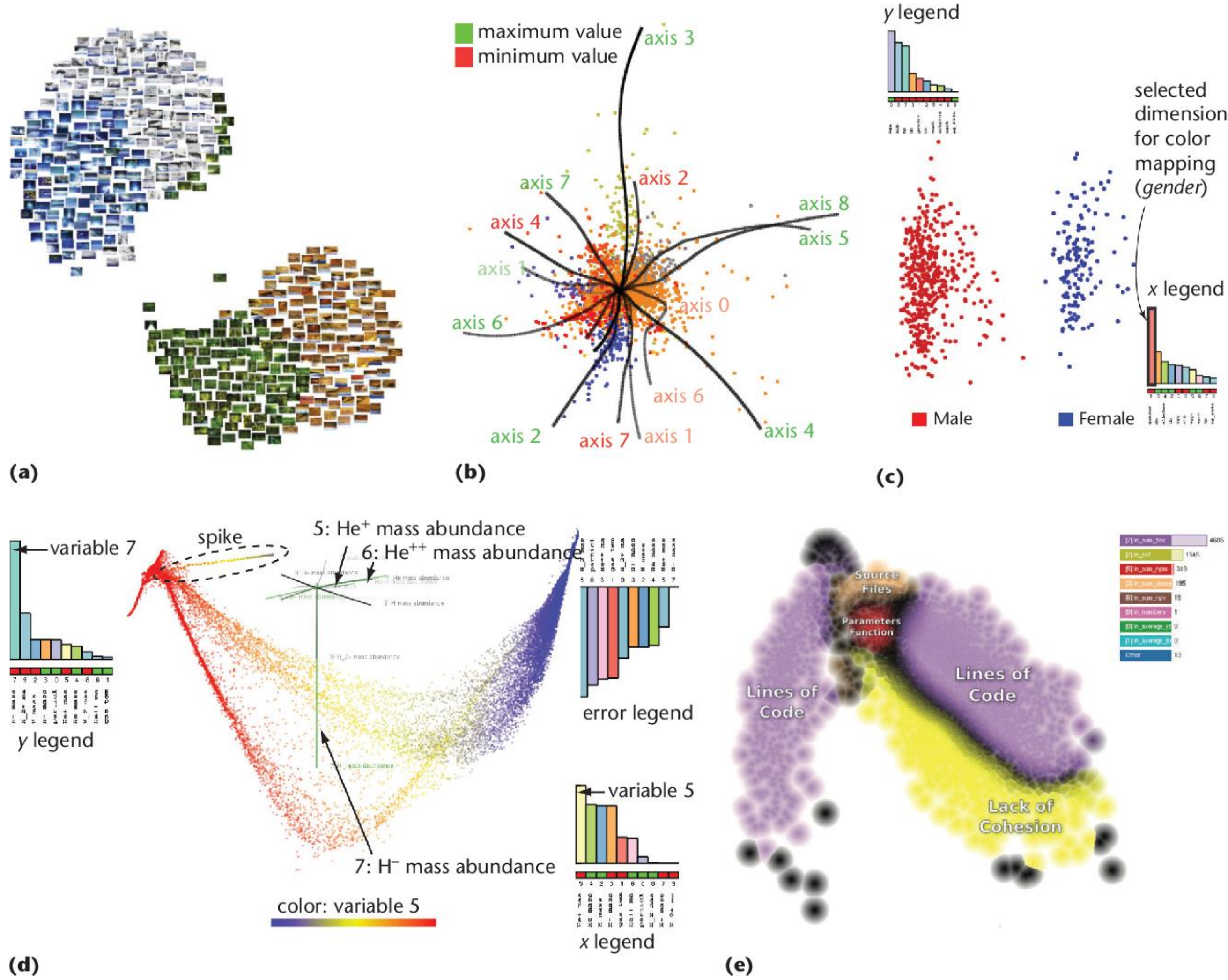


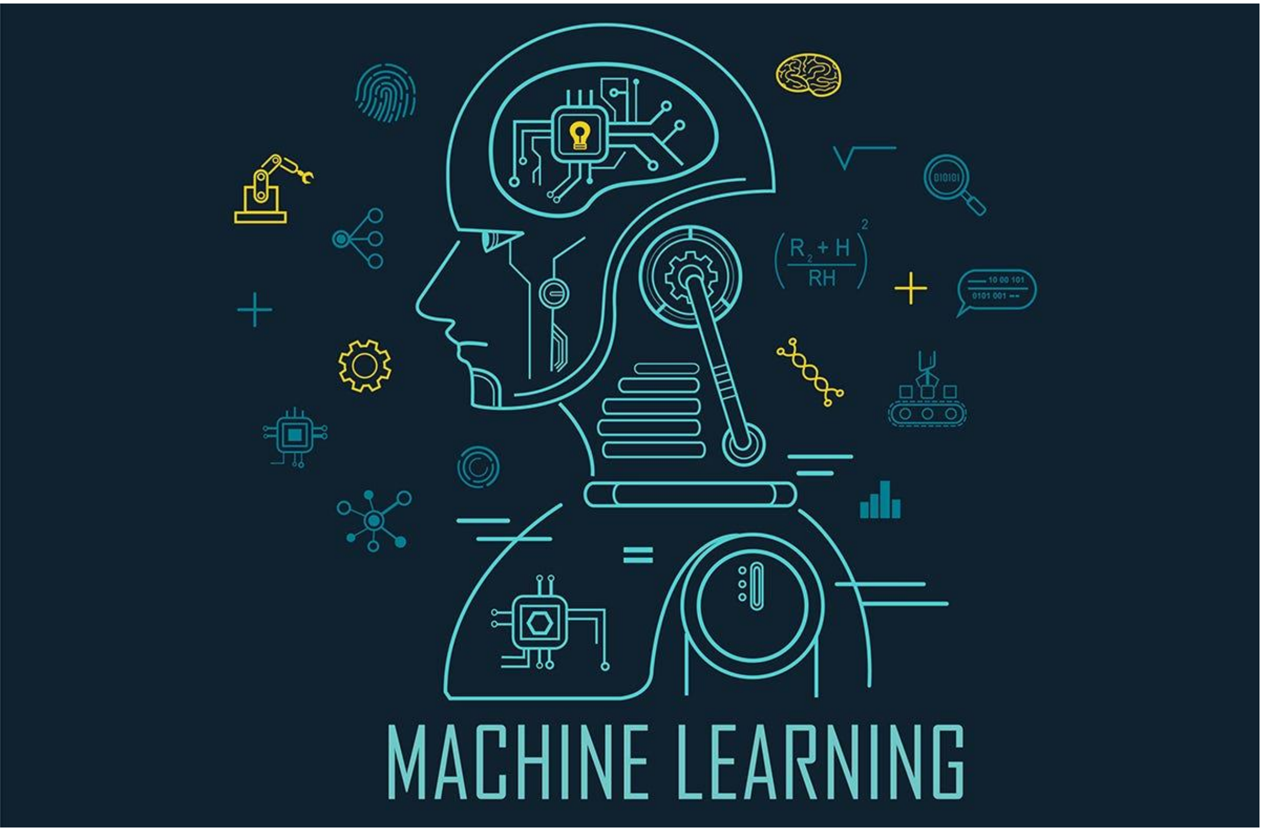
PLS direction (solid line) and first PCR direction (dotted line) are shown.
Trying to represent more population.

- More features, even (or especially) useless ones, increases our models ability to memorise the dataset, it doesn't need to find a general rule / relationship.
- This means overfitting and poorer out of sample performance.



- Let’s take a look in the lab.

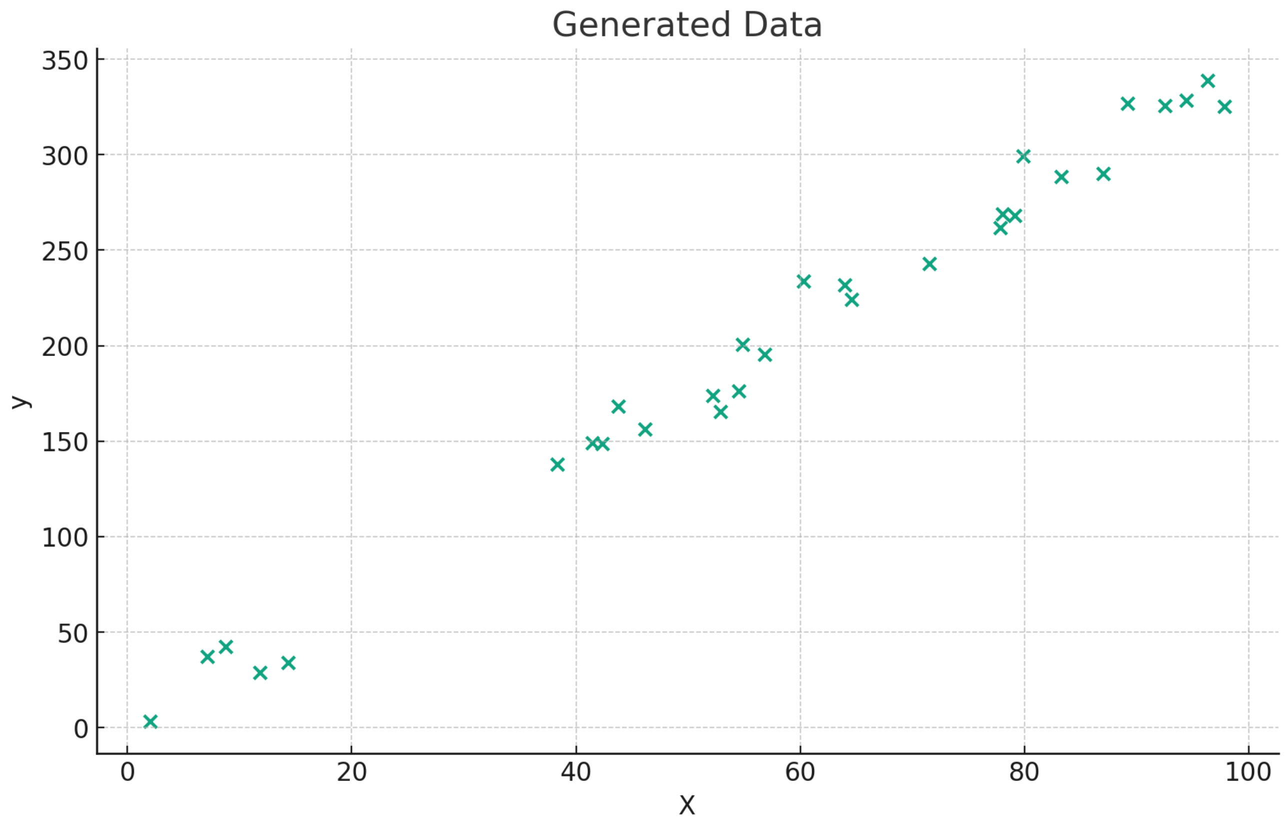




More regression

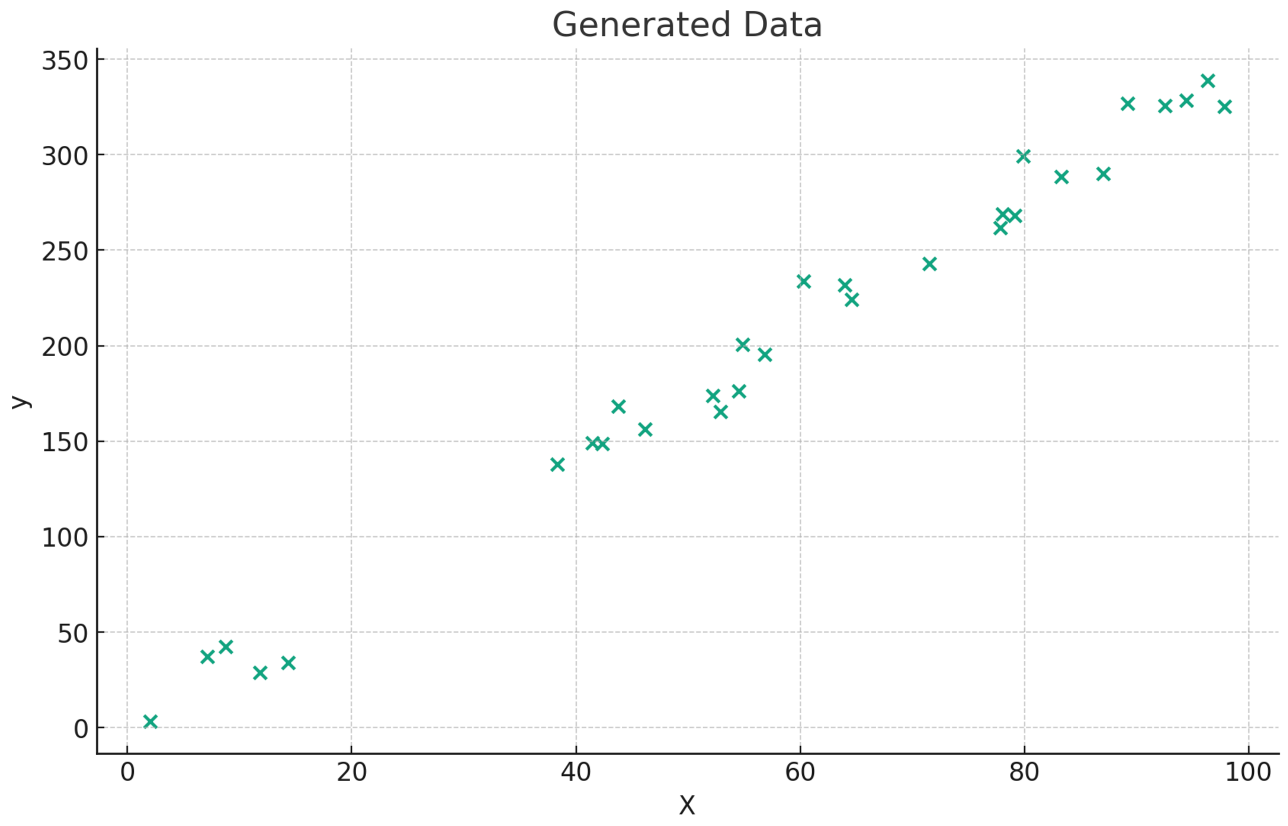
We will see later some machine learning approaches to improve of linear regression and help us with some of the issues.

Next though we can start exploring other machine learning approaches to regression.

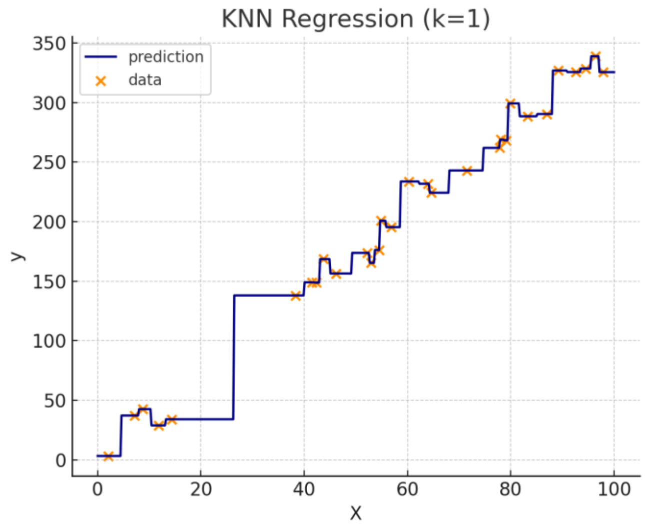


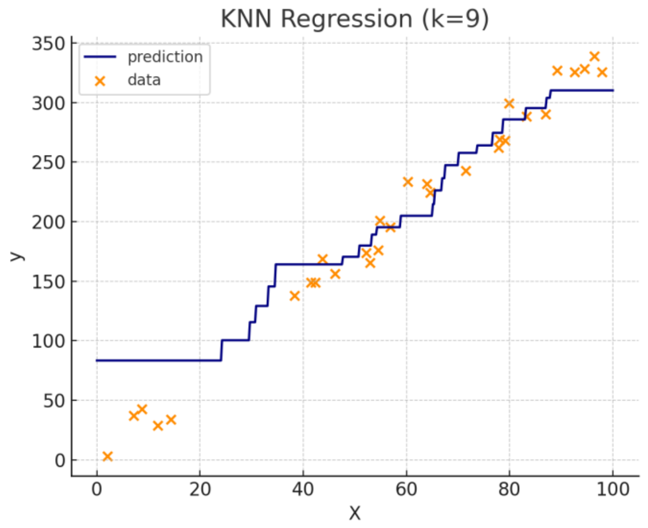
We can start exploring other machine learning approaches to regression.

- For example it's reasonably straight forward to extend the k nearest neighbour algorithm we have seen before.
- This is a non-parametric method, We don't make any assumptions about the distribution of the data or the relationship.

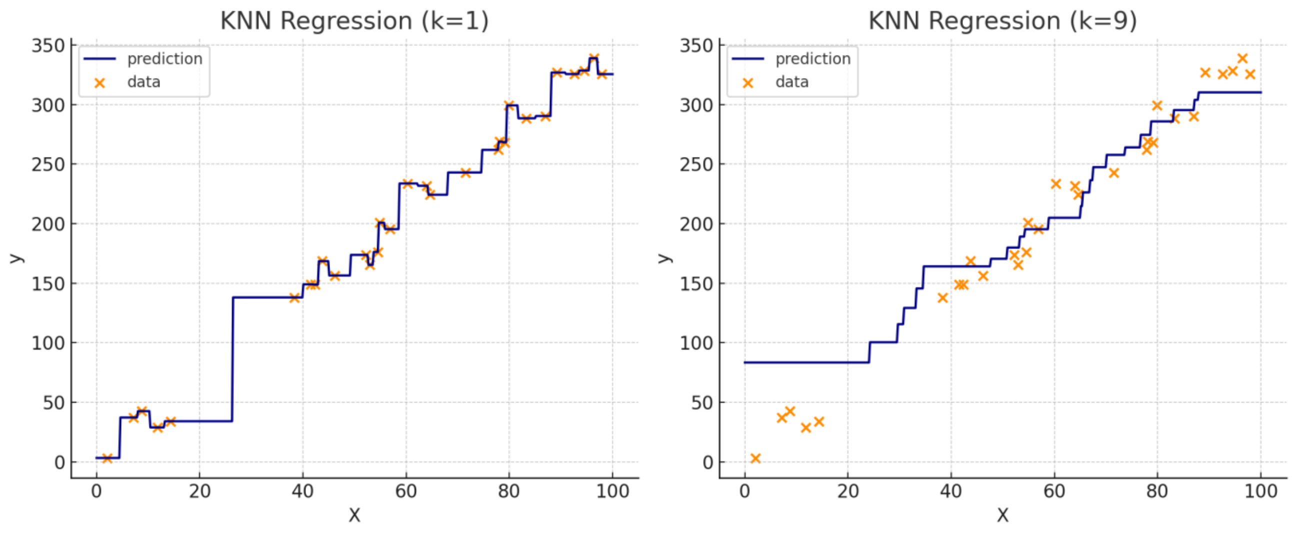


1. Choose the number 'k' of neighbors.
2. For a new data point, find the 'k' training data points that are closest to the point.
3. The predicted value is the average of the 'k' nearest neighbors' output values.

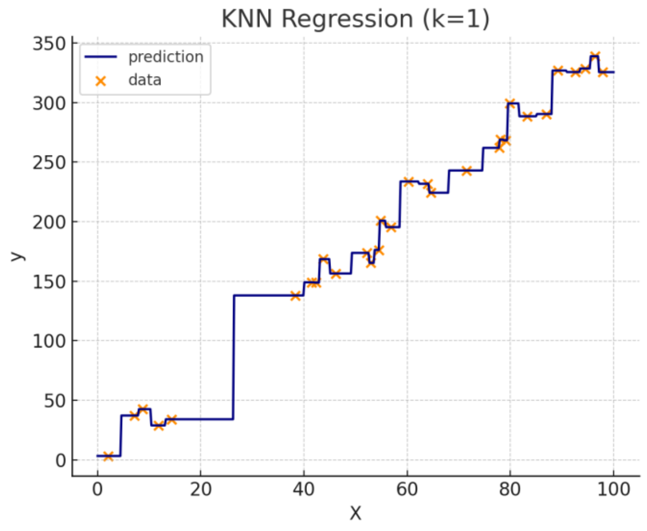


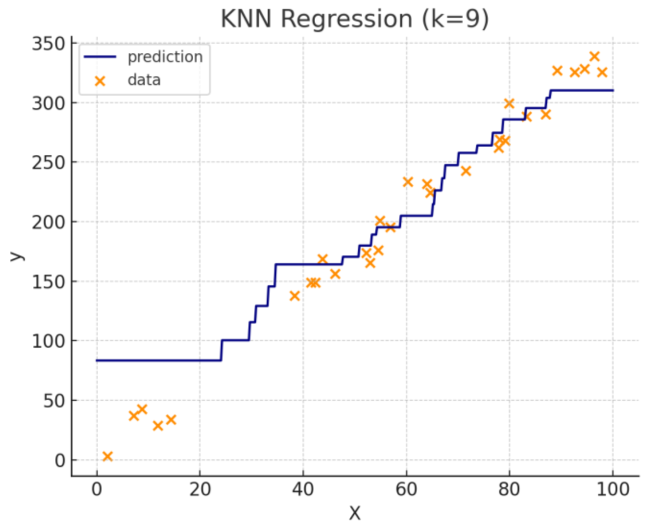


- No assumptions about the functional form of the model.
- Requires a distance metric (like Euclidean distance).
- The value of 'k' (number of neighbors) plays a crucial role: smaller values lead to noisy predictions, while larger values can smooth out the predictions excessively. **Bias – Variance trade-off**
- Computationally intensive for large datasets because it requires calculating the distance to each training instance.
- Sensitive to irrelevant or redundant features since all features contribute equally to the calculation of distances.

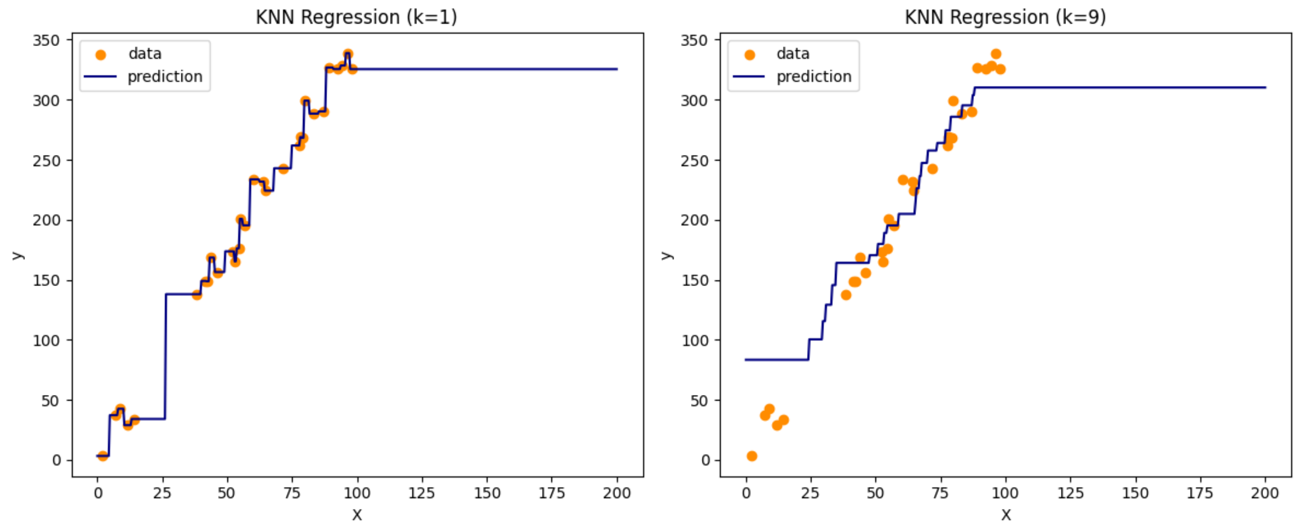


What would you predict for x = 200?





What would you predict for x = 200?



- The value of 'k' (number of neighbors) plays a crucial role: smaller values lead to noisy predictions, while larger values can smooth out the predictions excessively. **Bias – Variance trade-off**

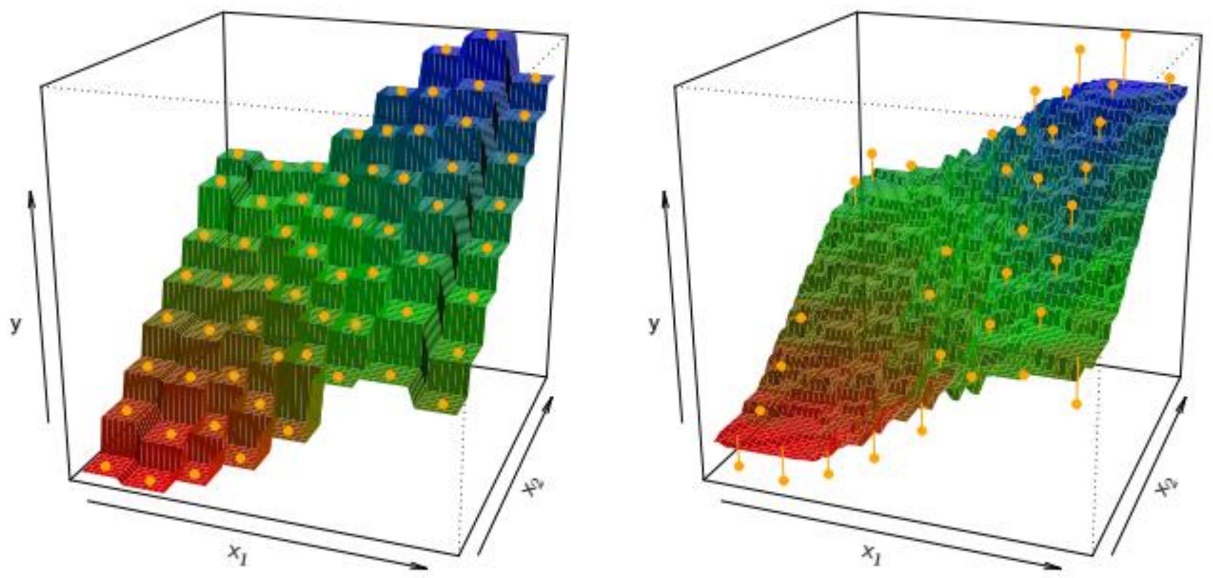
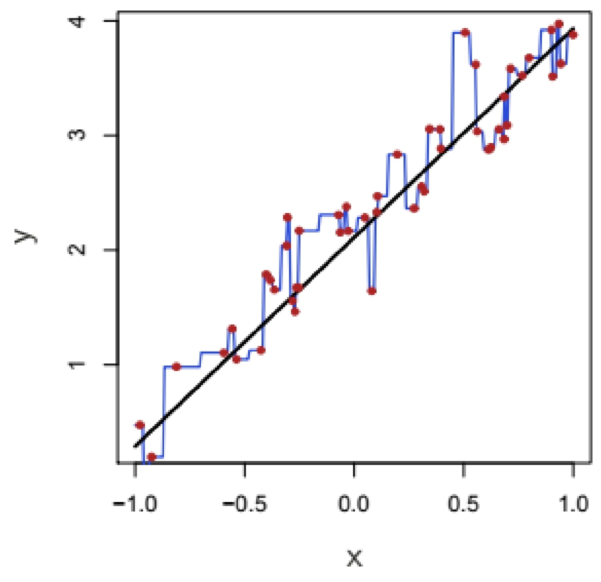
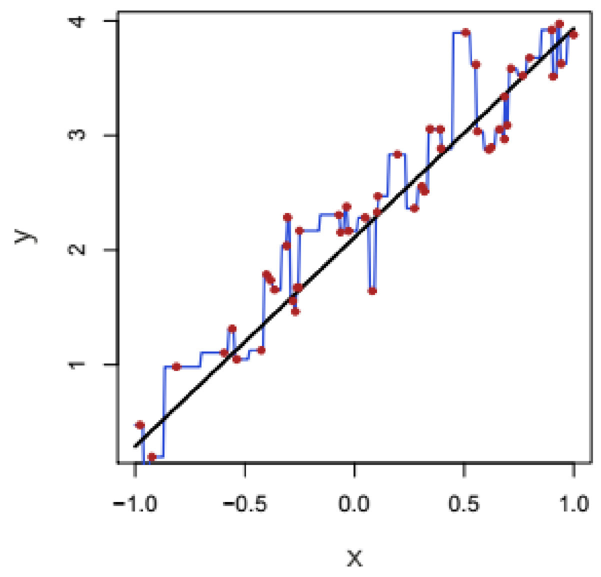


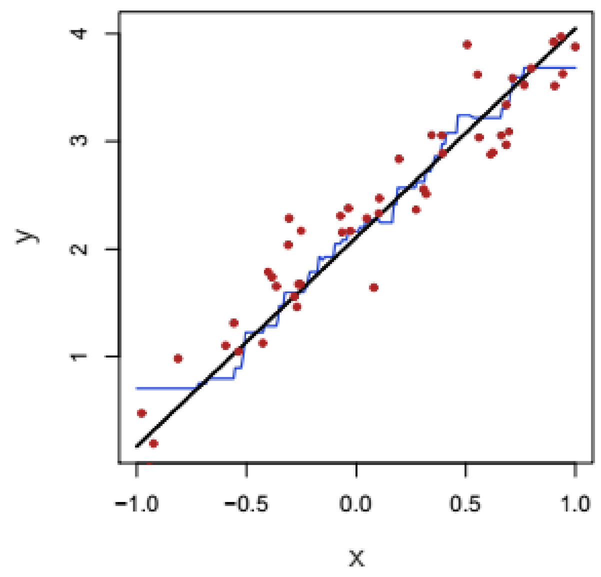
FIGURE 3.16. *Plots of $\hat{f}(X)$ using KNN regression on a two-dimensional data set with 64 observations (orange dots).* Left: *$K = 1$ results in a rough step function fit.* Right: *$K = 9$ produces a much smoother fit.*

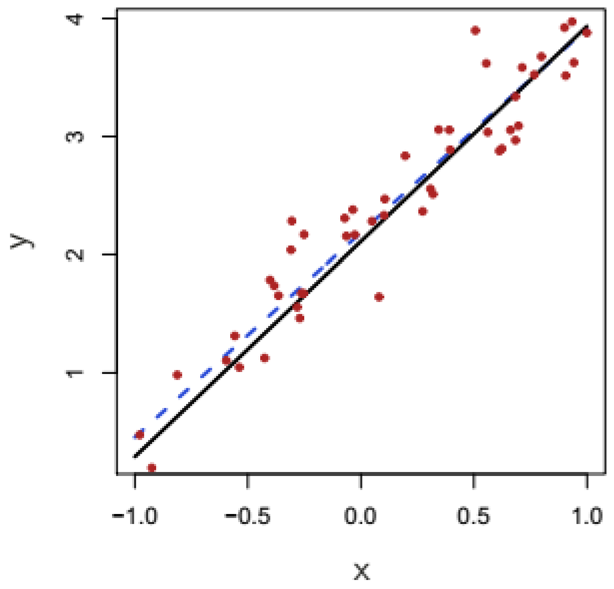


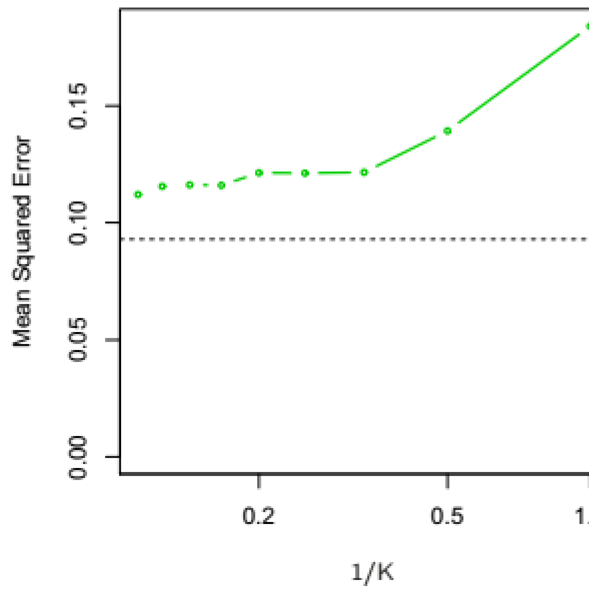
- The value of 'k' (number of neighbors) plays a crucial role: smaller values lead to noisy predictions, while larger values can smooth out the predictions excessively. **Bias – Variance trade-off**

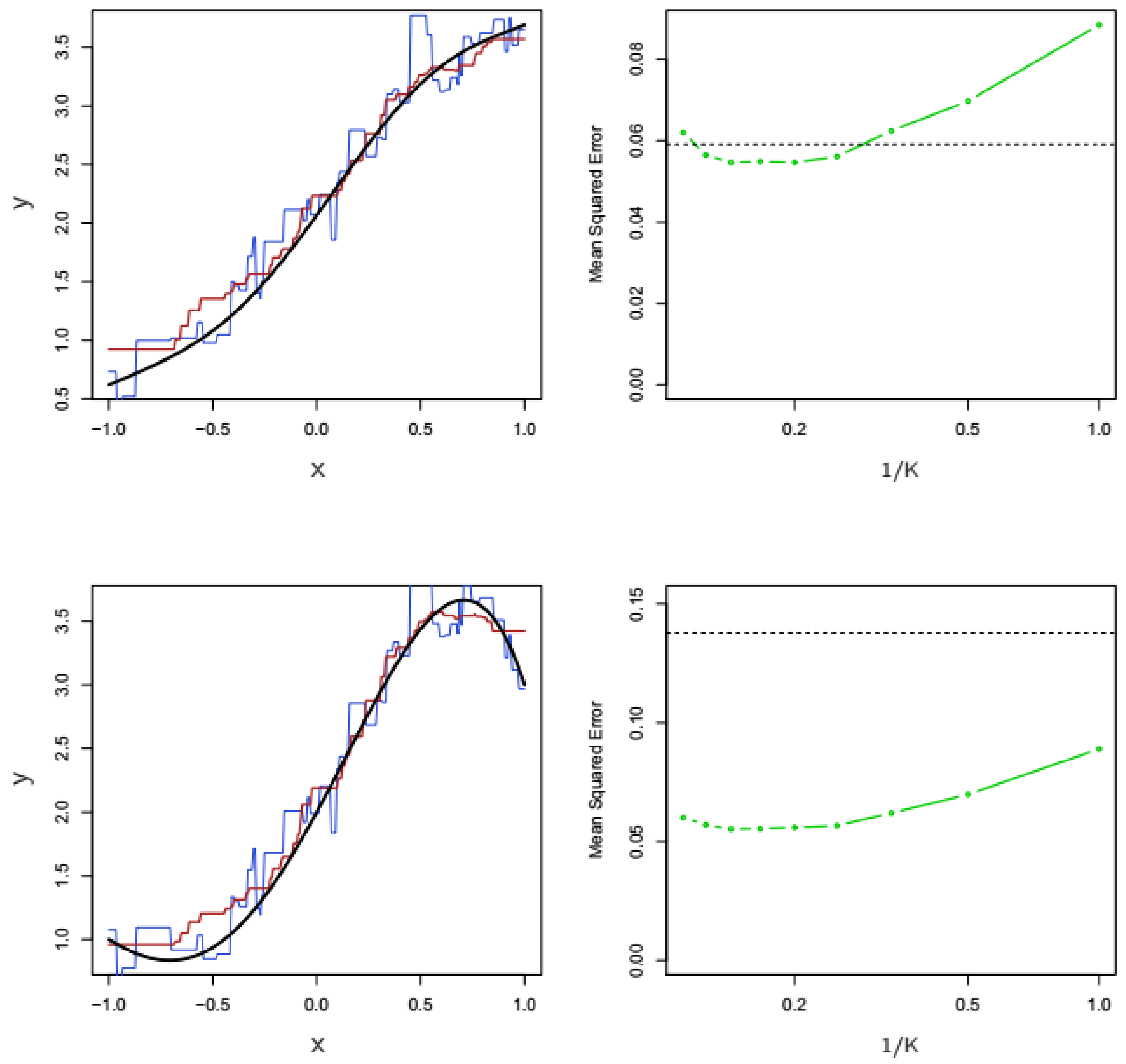
- The value of 'k' (number of neighbors) plays a crucial role: smaller values lead to noisy predictions, while larger values can smooth out the predictions excessively. **Bias – Variance trade-off**











- No assumptions about the functional form of the model.

As the number of features (dimensions) increases, the volume of the feature space increases exponentially.

This means that data points become sparser and, consequently, the distance between points tends to become more uniform. In other words, in high-dimensional spaces, most points are "far away" from each other, making it harder for KNN to distinguish which neighbors are genuinely "nearest.“

Also increases computational cost.

Remember sensitive to feature scaling

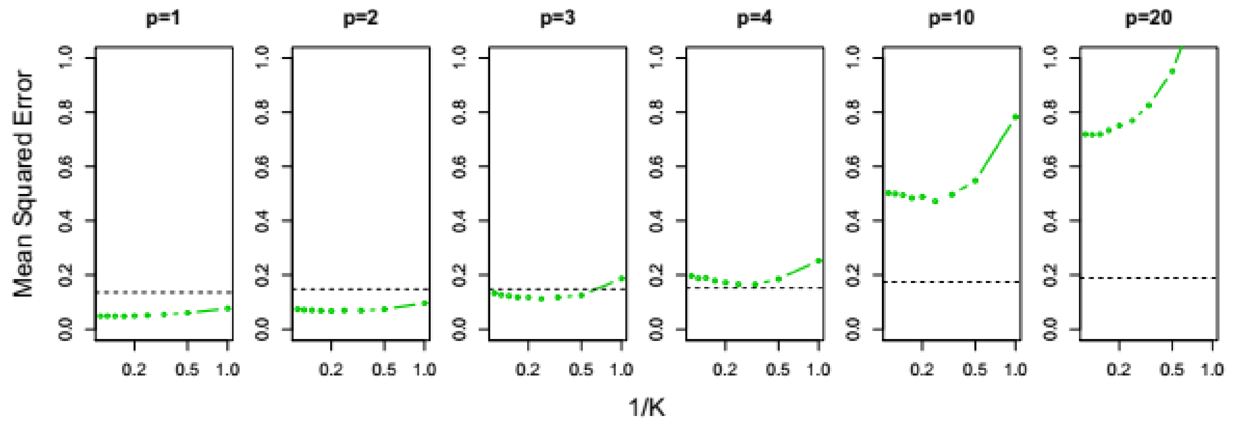


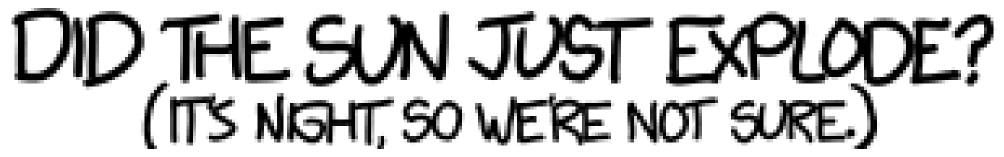
FIGURE 3.20. *Test MSE for linear regression (black dashed lines) and KNN (green curves) as the number of variables p increases. The true function is non-linear in the first variable, as in the lower panel in Figure 3.19, and does not depend on the additional variables. The performance of linear regression deteriorates slowly in the presence of these additional noise variables, whereas KNN's performance degrades much more quickly as p increases.*

Let’s take a look at another approach from our ML library. Bayesian Regression.

A quick degression on Bayesian thinking and approaches in ML.

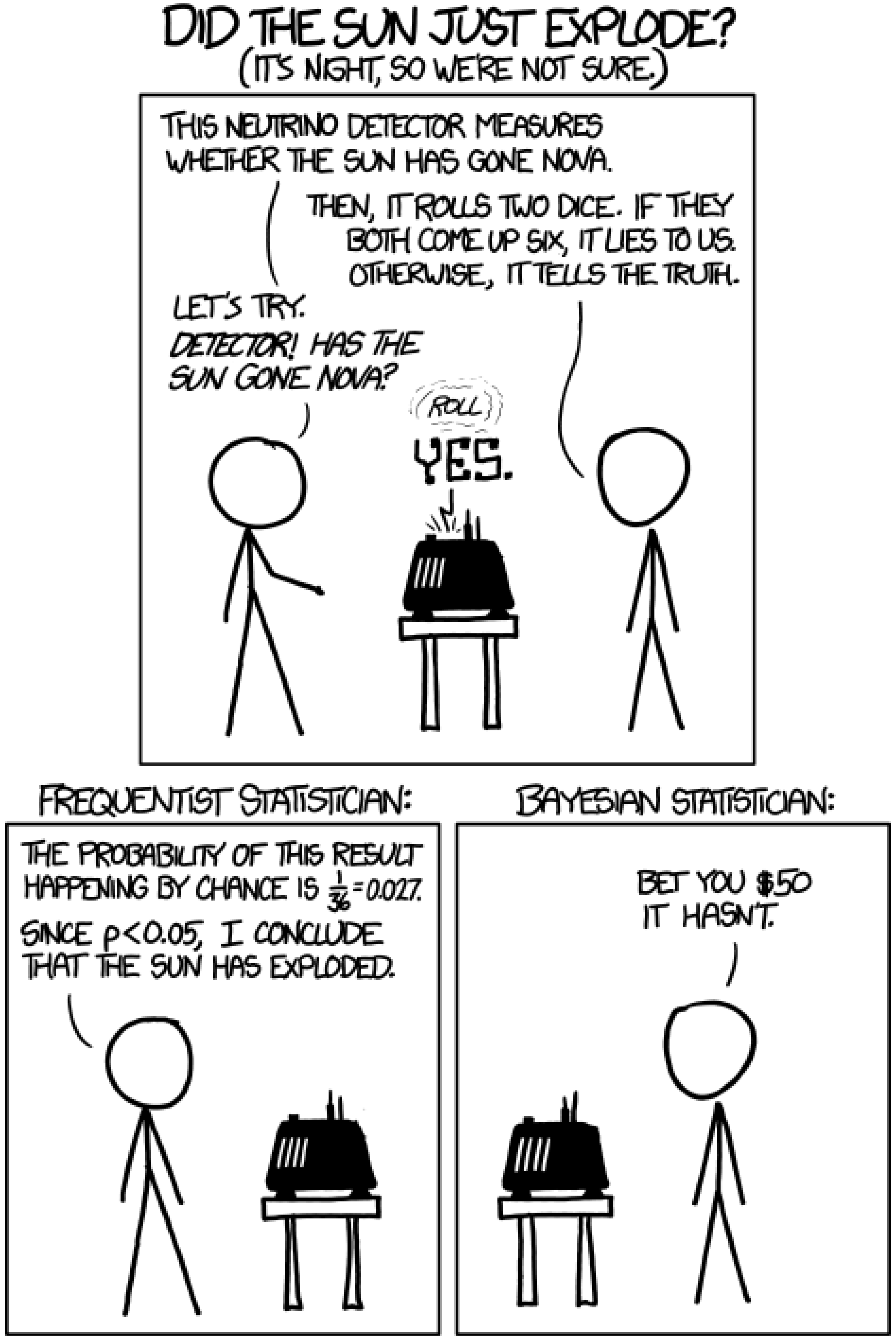
Let’s take a look at another approach from our ML library. Bayesian Regression.

A quick degression on Bayesian thinking and approaches in ML.



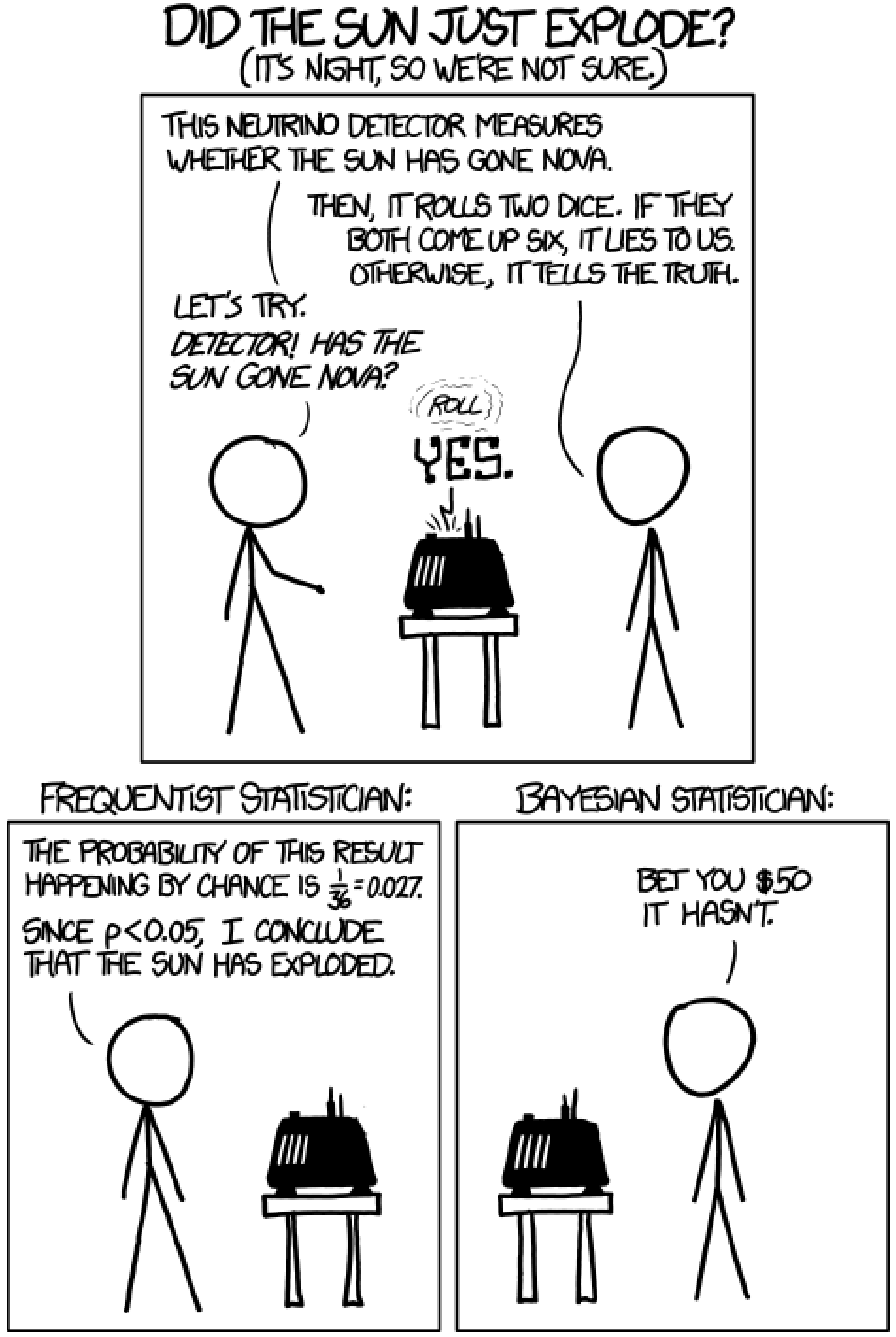
Let’s take a look at another approach from our ML library. Bayesian Regression.

A quick degression on Bayesian thinking and approaches in ML.


DID THE SUN JUST EXPLODE?
(IT'S NIGHT, SO WE'RE NOT SURE.)
THIS NEUTRINO DETECTOR MEASURES WHETHER THE SUN HAS GONE NOVA.
THEN, IT ROLLS TWO DICE. IF THEY BOTH COME UP SIX, IT LIES TO US. OTHERWISE, IT TELLS THE TRUTH.
LET'S TRY. DETECTOR! HAS THE SUN GONE NOVA?
(ROLL)
YES.
FREQUENTIST STATISTICIAN:
THE PROBABILITY OF THIS RESULT HAPPENING BY CHANCE IS 1/36 = 0.027. SINCE p < 0.05, I CONCLUDE THAT THE SUN HAS EXPLODED.
BAYESIAN STATISTICIAN:
BET YOU $50 IT HASN'T.

I have misplaced my phone somewhere in the home. I can use the phone locator on the base of the instrument to locate the phone and when I press the phone locator the phone starts beeping.

Problem: Which area of my home should I search?


DID THE SUN JUST EXPLODE?
(IT'S NIGHT, SO WE'RE NOT SURE.)
THIS NEUTRINO DETECTOR MEASURES WHETHER THE SUN HAS GONE NOVA.
THEN, IT ROLLS TWO DICE. IF THEY BOTH COME UP SIX, IT LIES TO US. OTHERWISE, IT TELLS THE TRUTH.
LET'S TRY. DETECTOR! HAS THE SUN GONE NOVA?
(ROLL)
YES.
FREQUENTIST STATISTICIAN:
THE PROBABILITY OF THIS RESULT HAPPENING BY CHANCE IS 1/36 = 0.027. SINCE p<0.05, I CONCLUDE THAT THE SUN HAS EXPLODED.
BAYESIAN STATISTICIAN:
BET YOU $50 IT HASN'T.

Frequentist Reasoning –

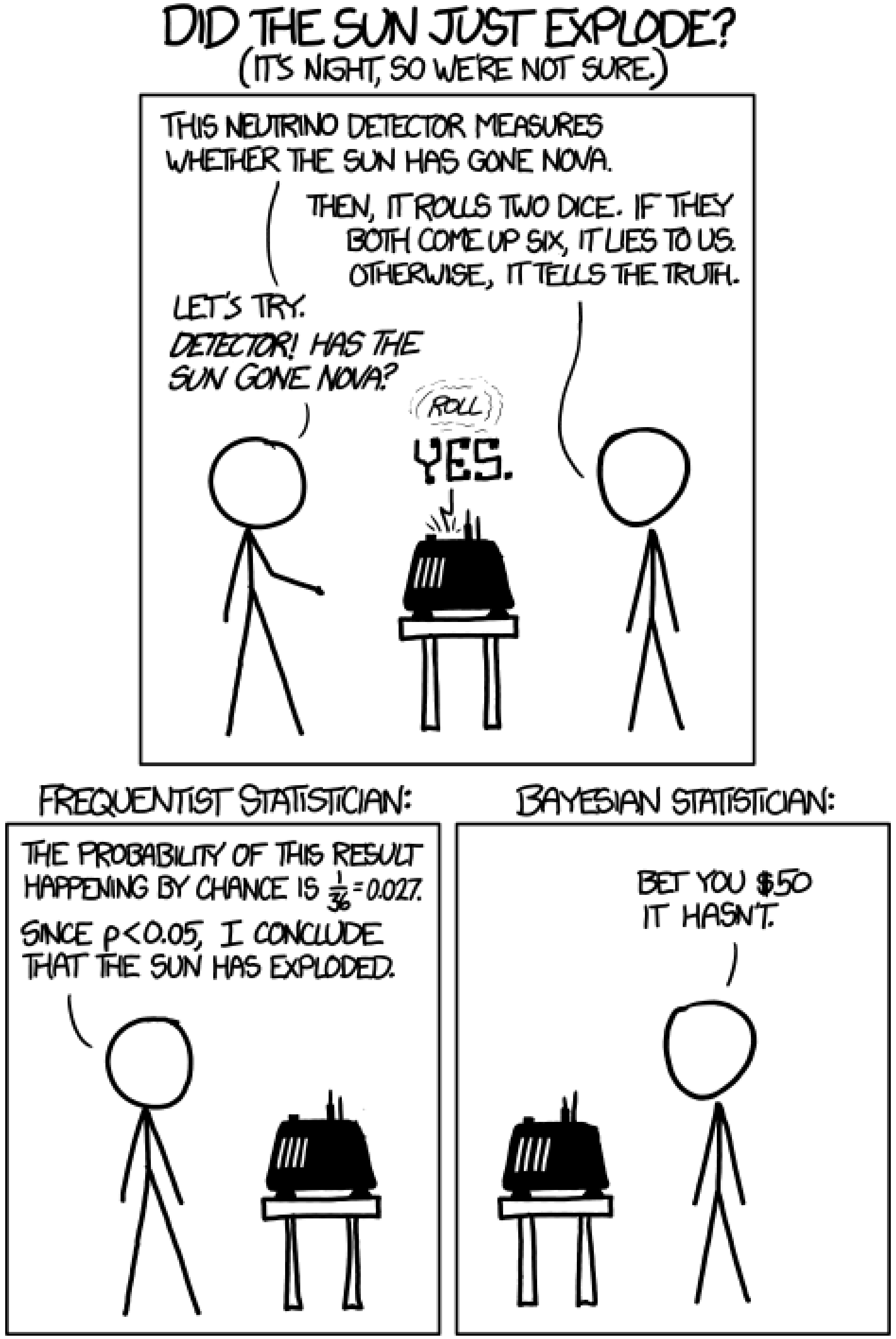
- I can hear the phone beeping.
- I have a mental model which helps me identify the area from which the sound is coming.

Therefore, upon hearing the beep, I infer the area of my home I must search to locate the phone.

Bayesian Reasoning –

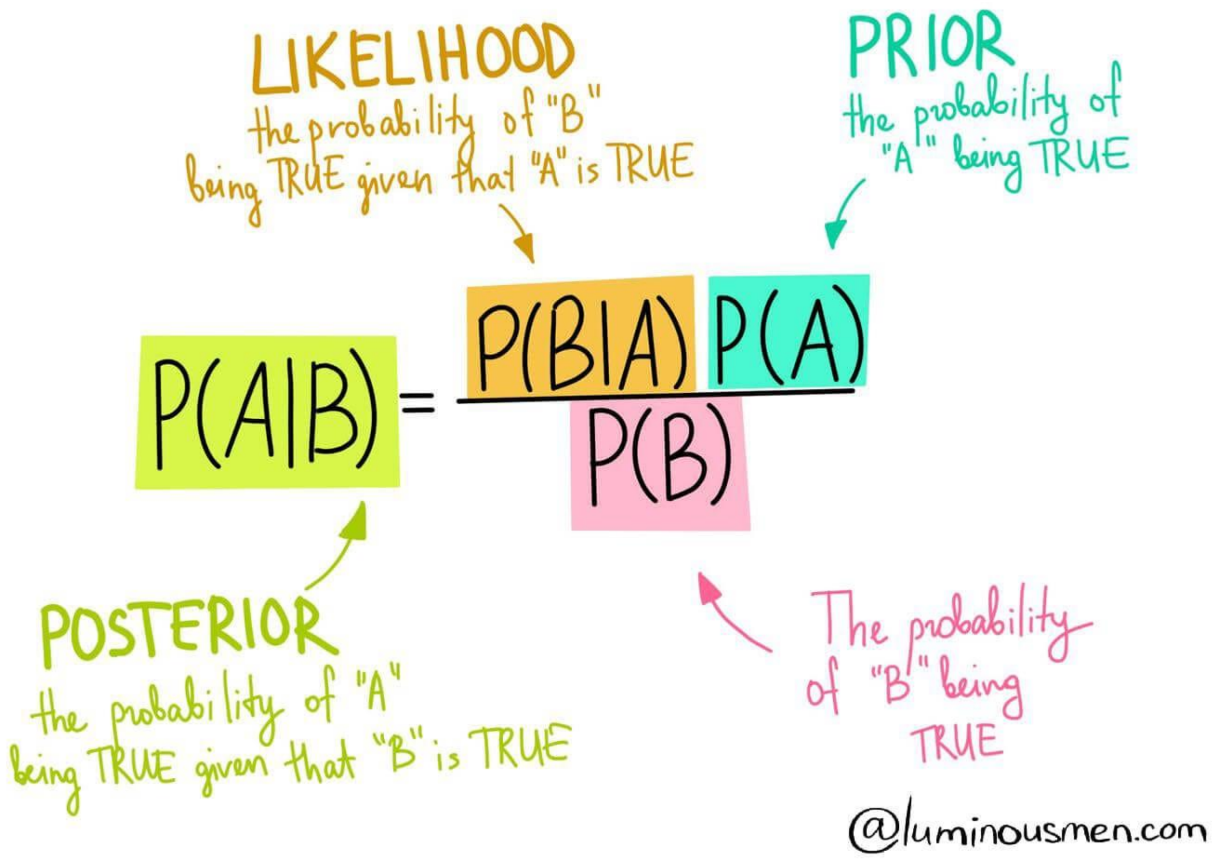
- I can hear the phone beeping.
- I have a mental model which helps me identify the area from which the sound is coming from.
- I also know the locations where I have misplaced the phone in the past. Or maybe I know the rooms I have been in today.

So, I combine my inferences using the beeps and my prior information about the locations I have misplaced the phone in the past to identify an area I must search to locate the phone.

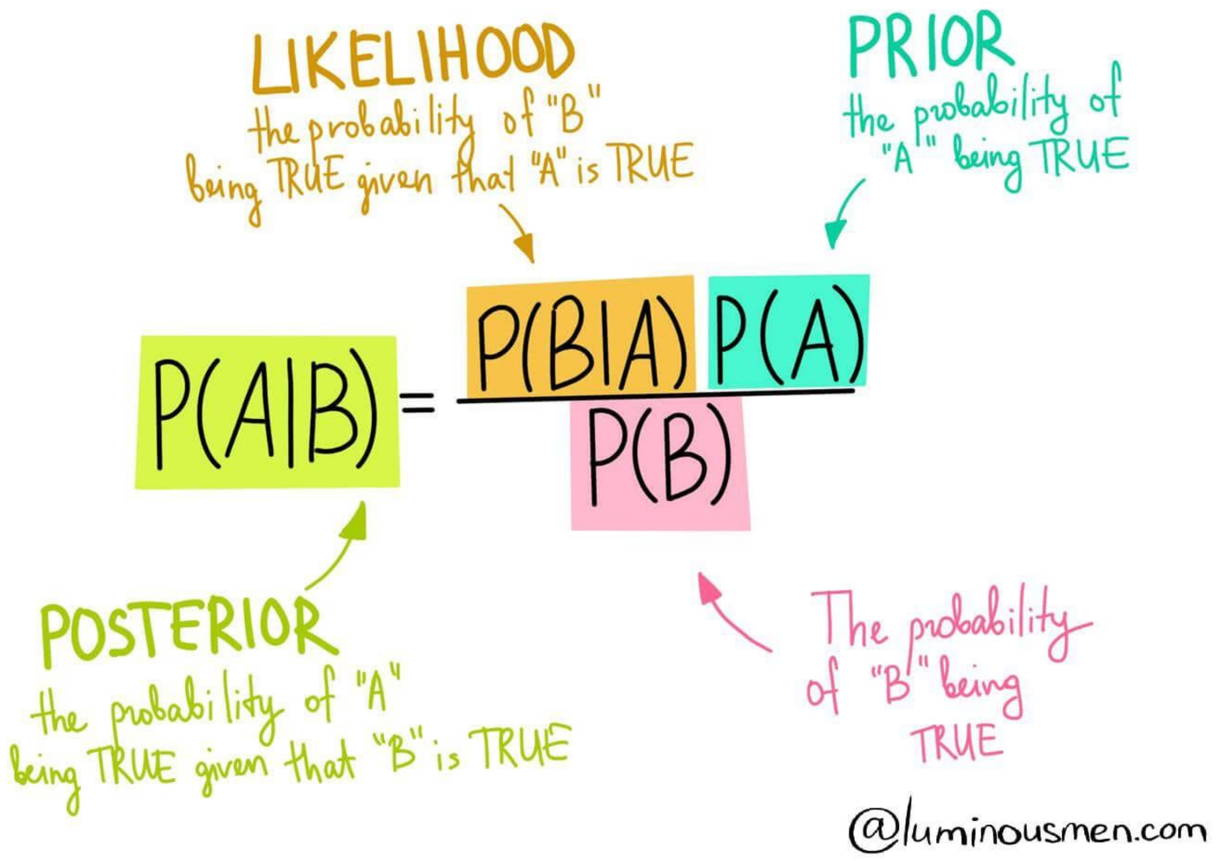

DID THE SUN JUST EXPLODE?
(IT'S NIGHT, SO WE'RE NOT SURE.)
THIS NEUTRINO DETECTOR MEASURES WHETHER THE SUN HAS GONE NOVA.
THEN, IT ROLLS TWO DICE. IF THEY BOTH COME UP SIX, IT LIES TO US. OTHERWISE, IT TELLS THE TRUTH.
LET'S TRY. DETECTOR! HAS THE SUN GONE NOVA?
(ROLL)
YES.
FREQUENTIST STATISTICIAN:
THE PROBABILITY OF THIS RESULT HAPPENING BY CHANCE IS $\frac{1}{36}=0.027$. SINCE $p<0.05$, I CONCLUDE THAT THE SUN HAS EXPLODED.
BAYESIAN STATISTICIAN:
BET YOU $50 IT HASN'T.

Bayesian analysts formulate probabilistic statements about uncertain events before collecting any additional evidence (i.e., “data”).

These ex-ante probabilities (or, more generally, probability distributions plus underlying parameters) are called **priors**.

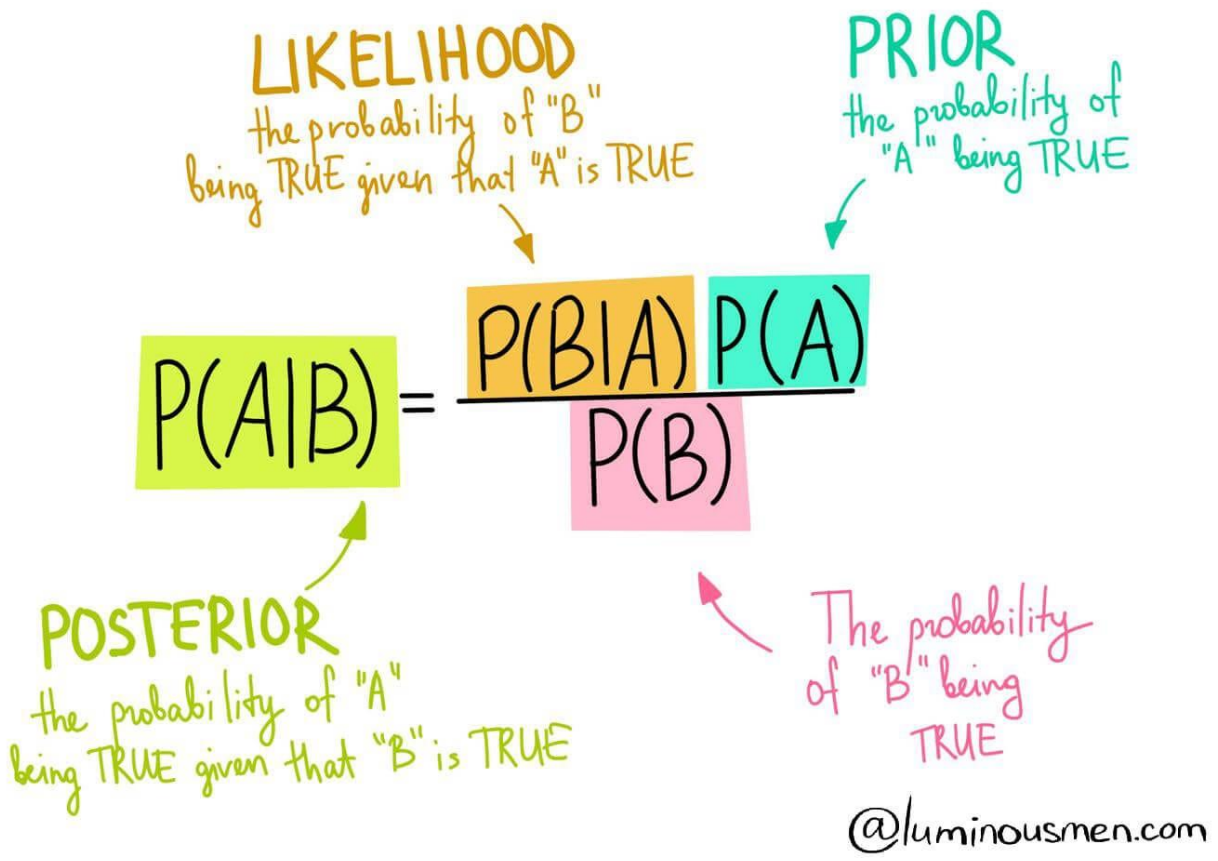

LIKELIHOOD
the probability of "B" being TRUE given that "A" is TRUE
PRIOR
the probability of "A" being TRUE
P(A|B) = P(B|A) P(A) / P(B)
POSTERIOR
the probability of "A" being TRUE given that "B" is TRUE
The probability of "B" being TRUE
@luminousmen.com

Bayesian modeling is not about point estimation of a parameter value, θ, but rather updating and sharpening our subjective beliefs (our "prior") about θ from the sample data. Thus, the sample data should contribute to "learning" about θ.


LIKELIHOOD
the probability of "B" being TRUE given that "A" is TRUE
PRIOR
the probability of "A" being TRUE
P(A|B) = P(B|A) P(A) / P(B)
POSTERIOR
the probability of "A" being TRUE given that "B" is TRUE
The probability of "B" being TRUE
@luminousmen.com

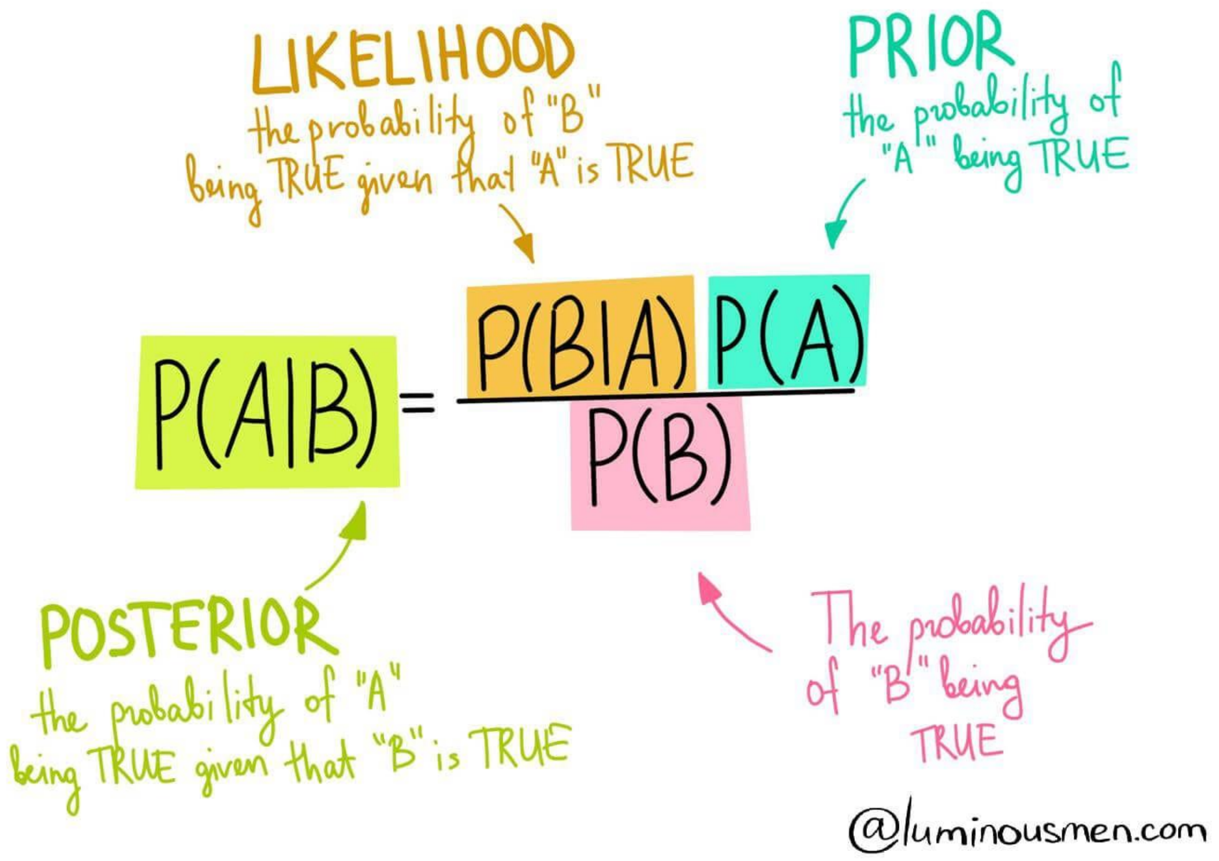
Bayes rule refresher:

https://youtu.be/4hHA-oqpNig?si=GwKpDiehrLCQBDMJ&t=219


LIKELIHOOD
the probability of "B" being TRUE given that "A" is TRUE
PRIOR
the probability of "A" being TRUE
$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}$
POSTERIOR
the probability of "A" being TRUE given that "B" is TRUE
The probability of "B" being TRUE
@luminousmen.com

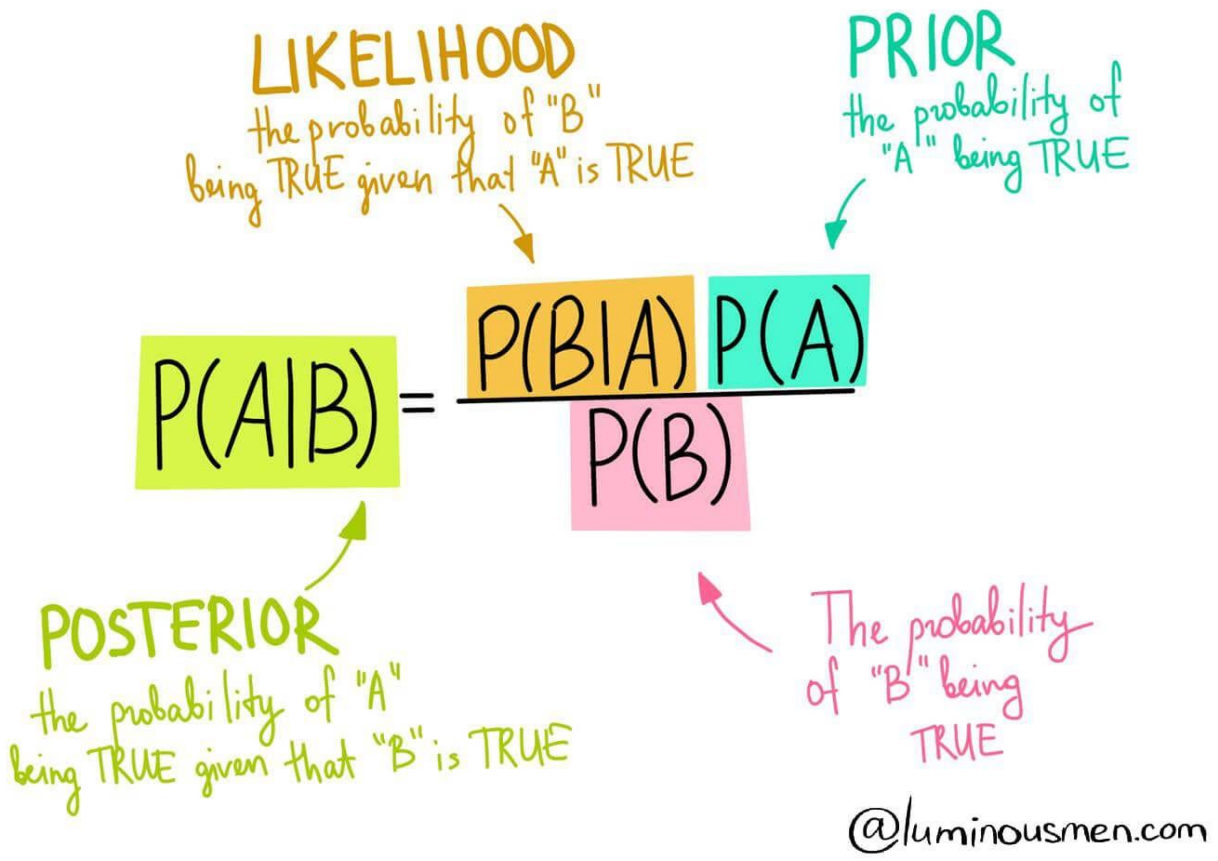
With lot's of data there isn't a big distinction. If the sample size is large and the likelihood function "well-behaved" (which usually means a simple function with a clear maximum, plus a small dimension for θ), classical and Bayesian analysis are essentially on the same footing and will produce virtually identical results.

This is because the likelihood function and empirical data will dominate any prior assumptions in the Bayesian approach.

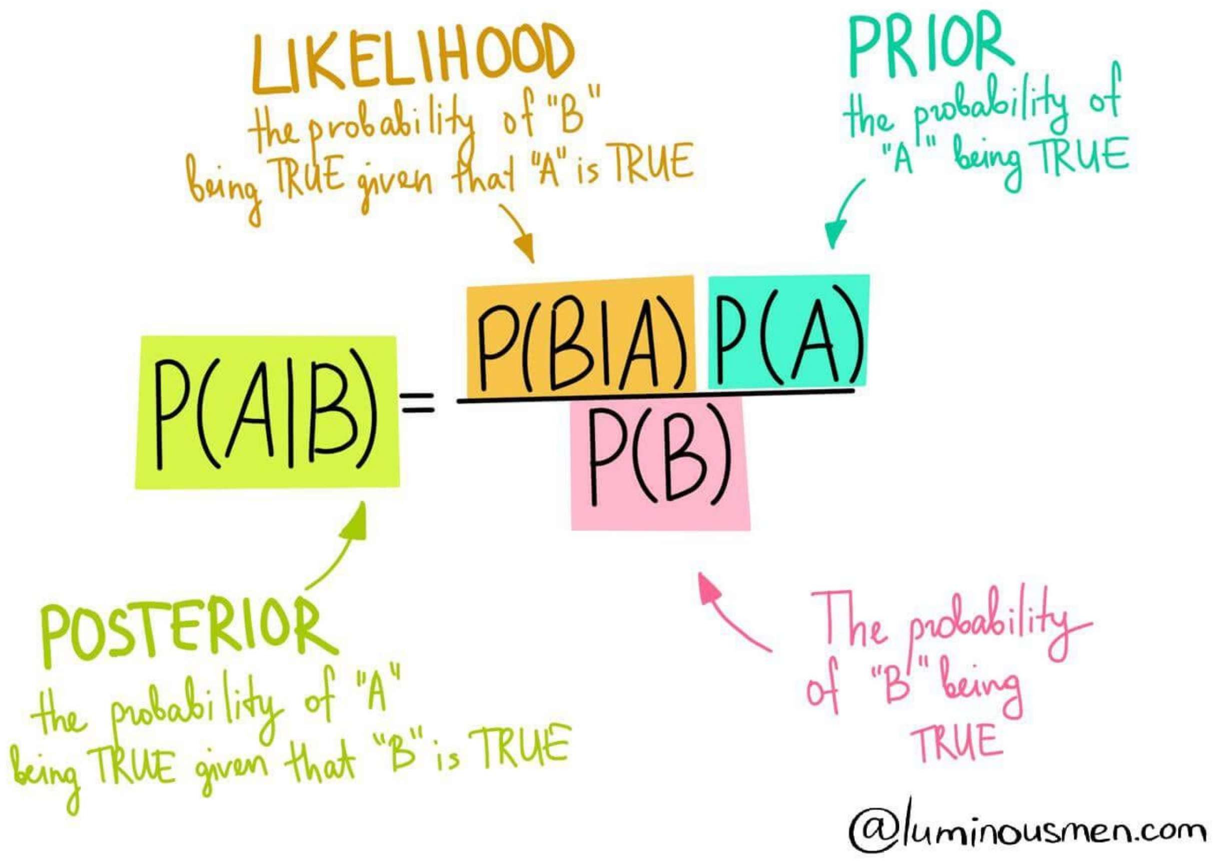

LIKELIHOOD
the probability of "B" being TRUE given that "A" is TRUE
PRIOR
the probability of "A" being TRUE
P(A|B) = P(B|A) P(A) / P(B)
POSTERIOR
the probability of "A" being TRUE given that "B" is TRUE
The probability of "B" being TRUE
@luminousmen.com

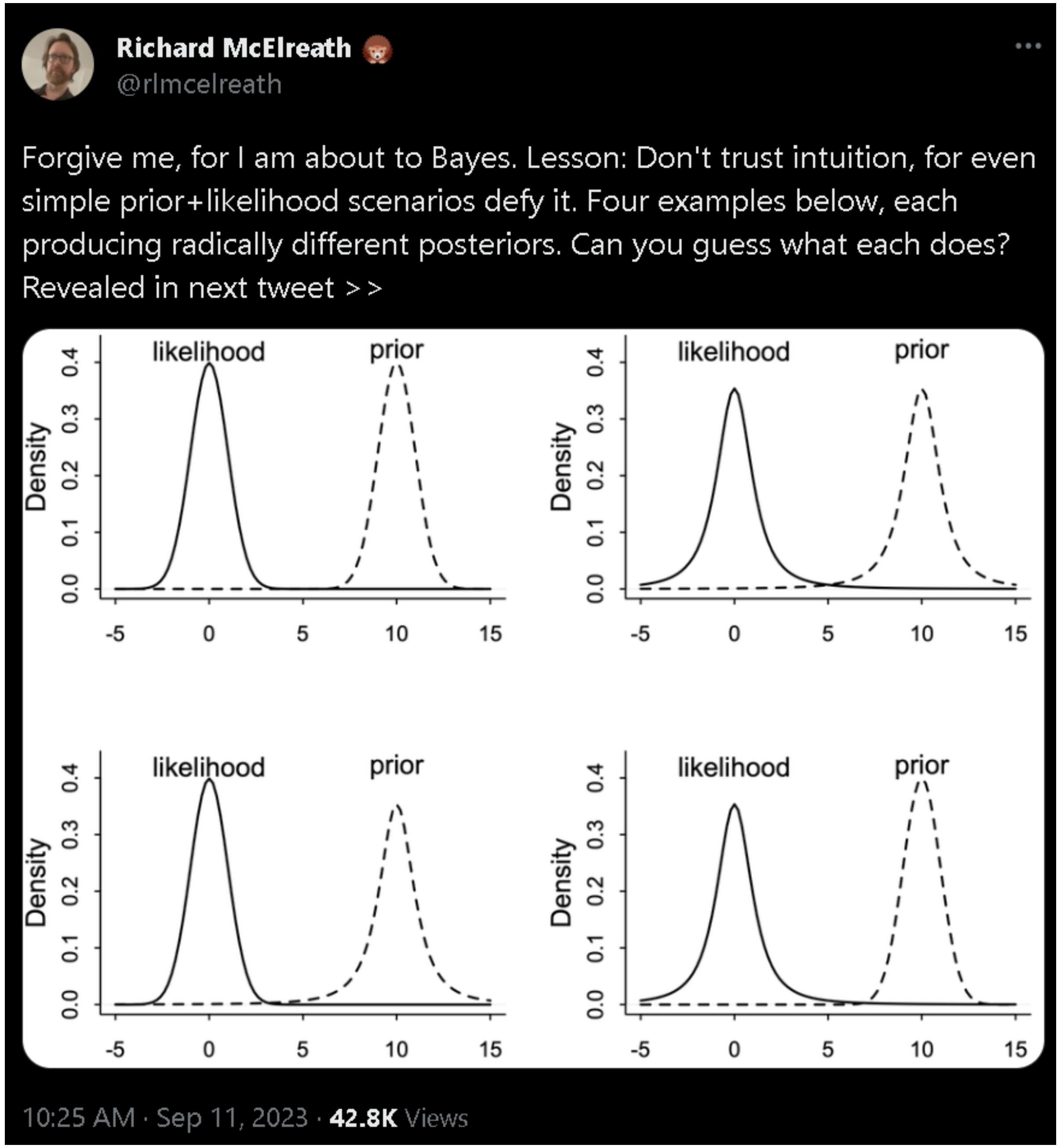
Bayesian approaches “help” us in small sample situations.

- Bayesian results do not depend on asymptotic theory to hold for their interpretability.
- Bayesian approach combines the sparse data with subjective priors. Well-informed priors can increase the accuracy and efficiency of the model.
- Conversely, of course, poorly chosen priors can produce misleading posterior inference in this case.


LIKELIHOOD
the probability of "B" being TRUE given that "A" is TRUE
PRIOR
the probability of "A" being TRUE
$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}$
POSTERIOR
the probability of "A" being TRUE given that "B" is TRUE
The probability of "B" being TRUE
@luminousmen.com

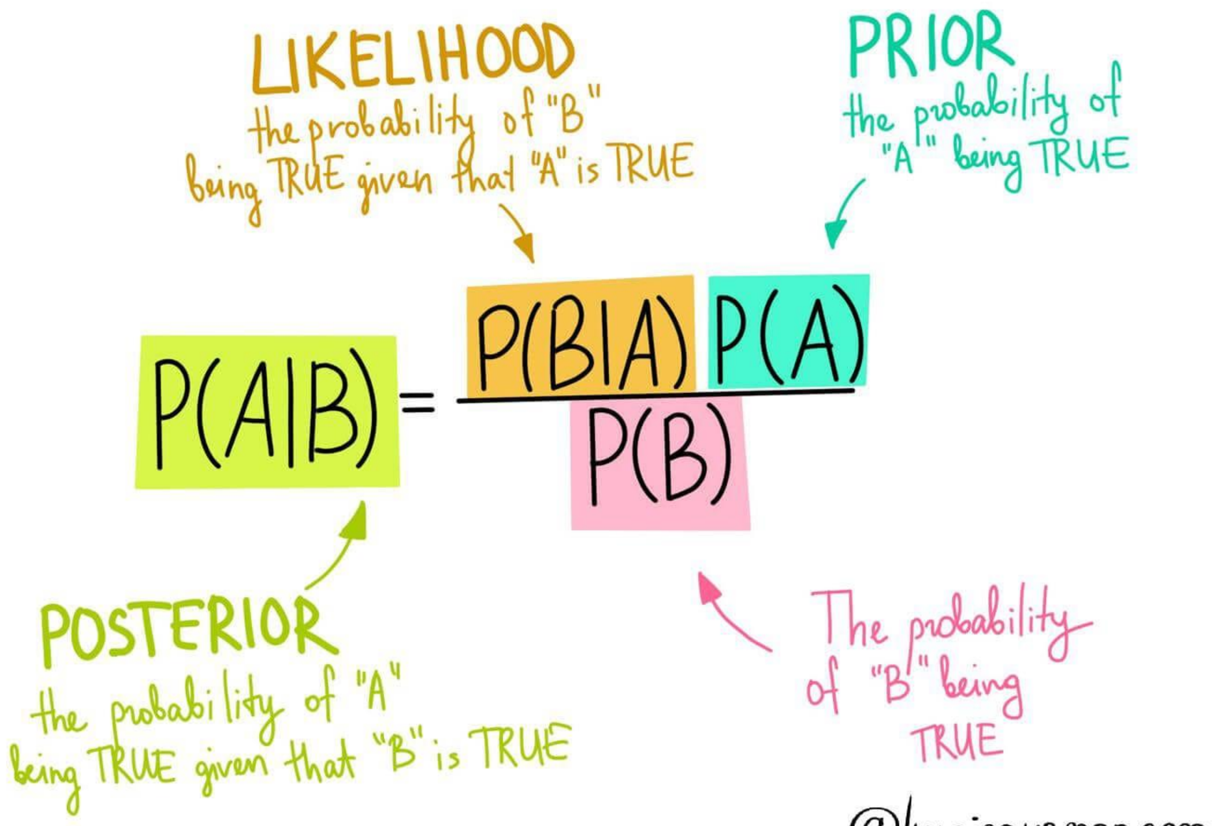
The choice between Bayesian and classical estimation often distills to **a choice between trusting the asymptotic properties of estimators and trusting one's priors**.


LIKELIHOOD
the probability of "B" being TRUE given that "A" is TRUE
PRIOR
the probability of "A" being TRUE
P(A|B) = P(B|A) P(A) / P(B)
POSTERIOR
the probability of "A" being TRUE given that "B" is TRUE
The probability of "B" being TRUE
@luminousmen.com


Richard McElreath @rlmcelreath

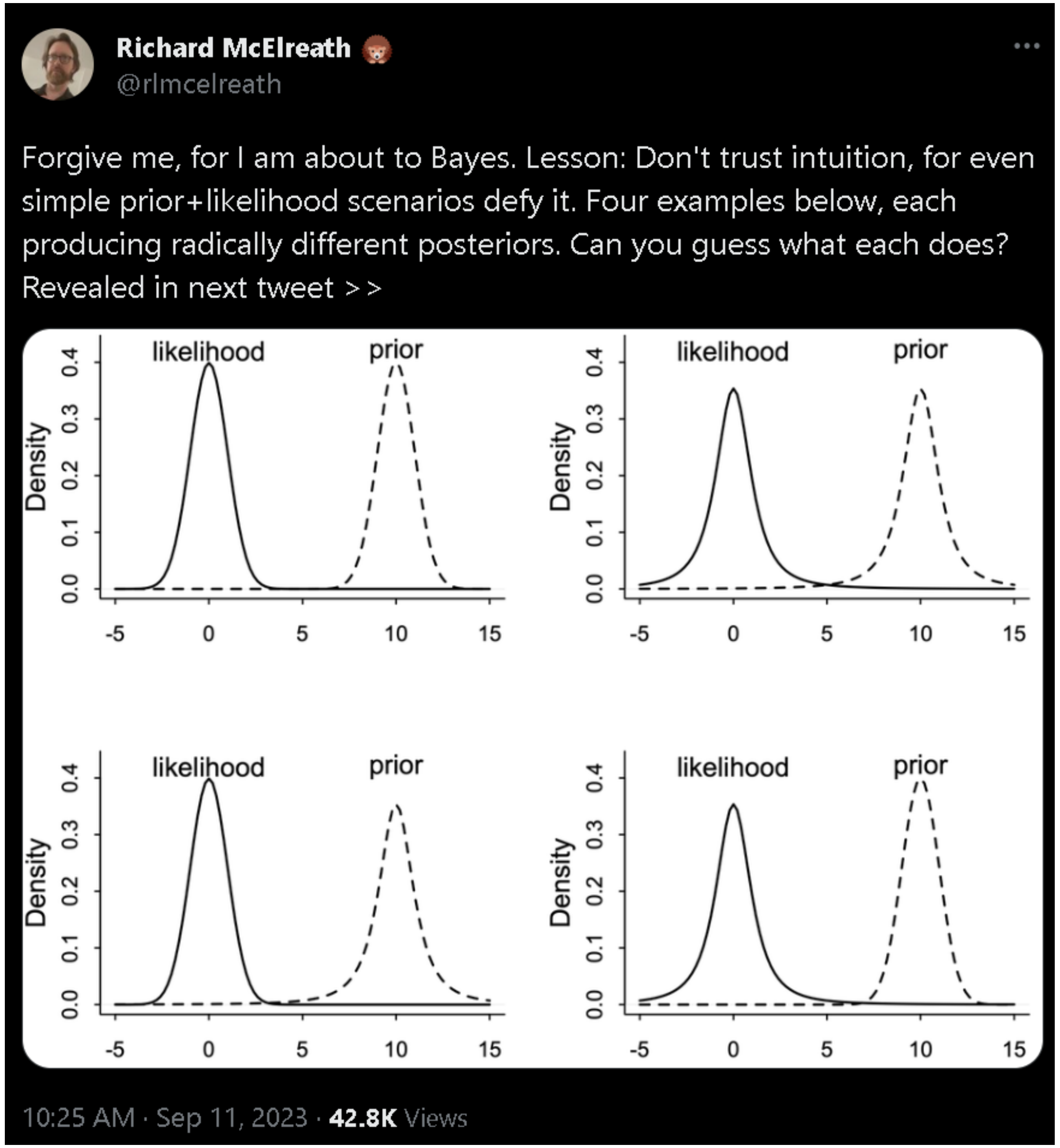
Forgive me, for I am about to Bayes. Lesson: Don't trust intuition, for even simple prior+likelihood scenarios defy it. Four examples below, each producing radically different posteriors. Can you guess what each does? Revealed in next tweet > >

10:25 AM · Sep 11, 2023 · 42.8K Views



LIKELIHOOD
the probability of "B" being TRUE given that "A" is TRUE
PRIOR
the probability of "A" being TRUE
POSTERIOR
the probability of "A" being TRUE given that "B" is TRUE
The probability of "B" being TRUE
@luminousmen.com

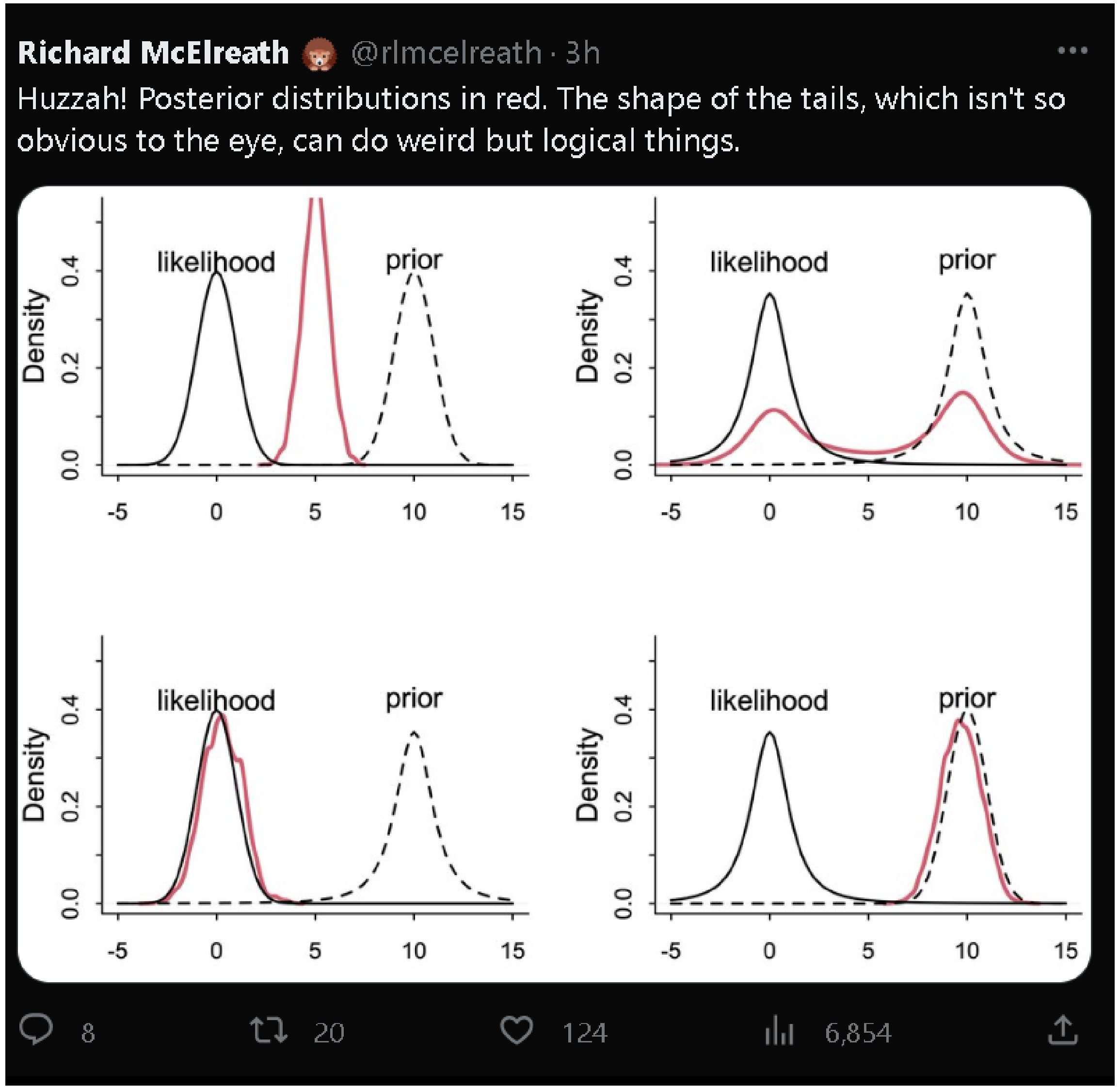
$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}$$



Richard McElreath 🦔
@rlmcelreath

Forgive me, for I am about to Bayes. Lesson: Don't trust intuition, for even simple prior+likelihood scenarios defy it. Four examples below, each producing radically different posteriors. Can you guess what each does? Revealed in next tweet > >

10:25 AM · Sep 11, 2023 · **42.8K** Views



Richard McElreath 🦔 @rlmcelreath · 3h

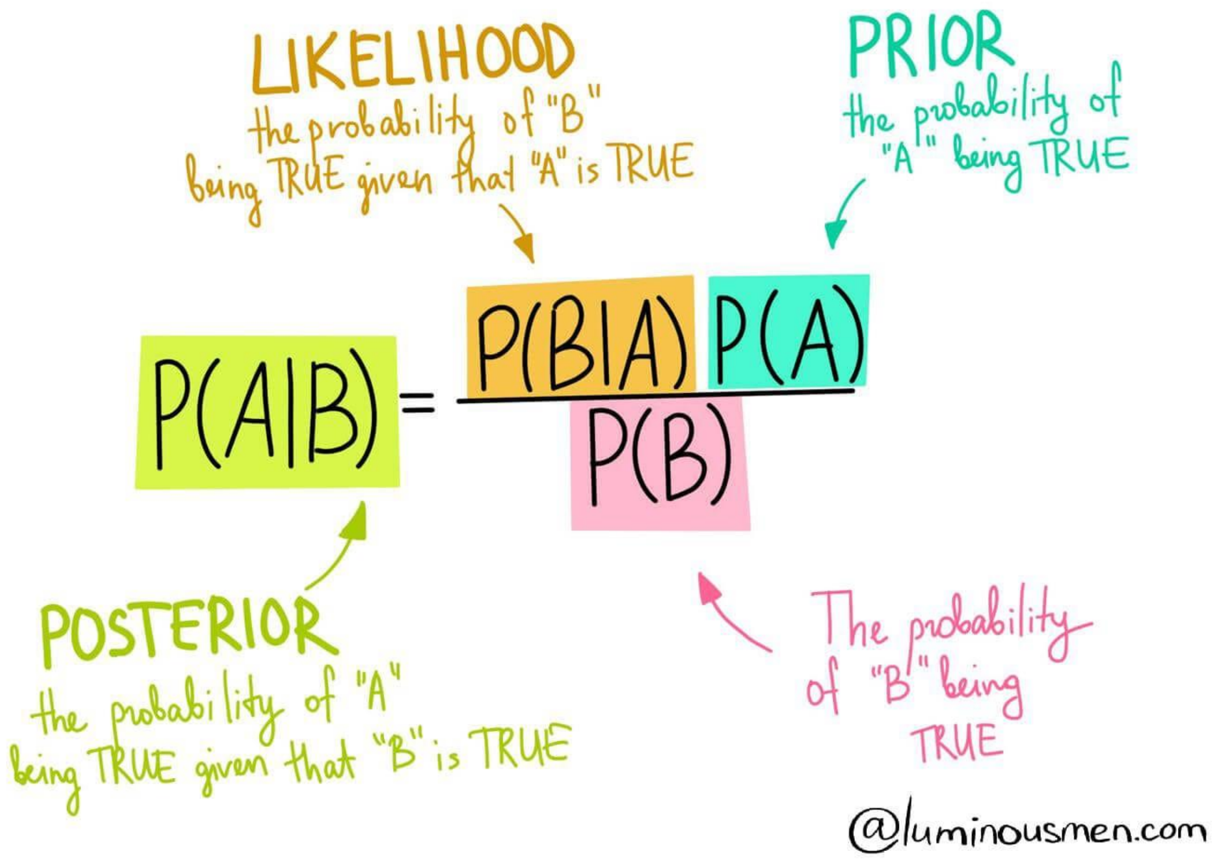
Huzzah! Posterior distributions in red. The shape of the tails, which isn't so obvious to the eye, can do weird but logical things.

8 20 124 6,854

In **Bayesian** Linear Regression we assume the responses are sampled from a probability distribution such as the normal (Gaussian) distribution.

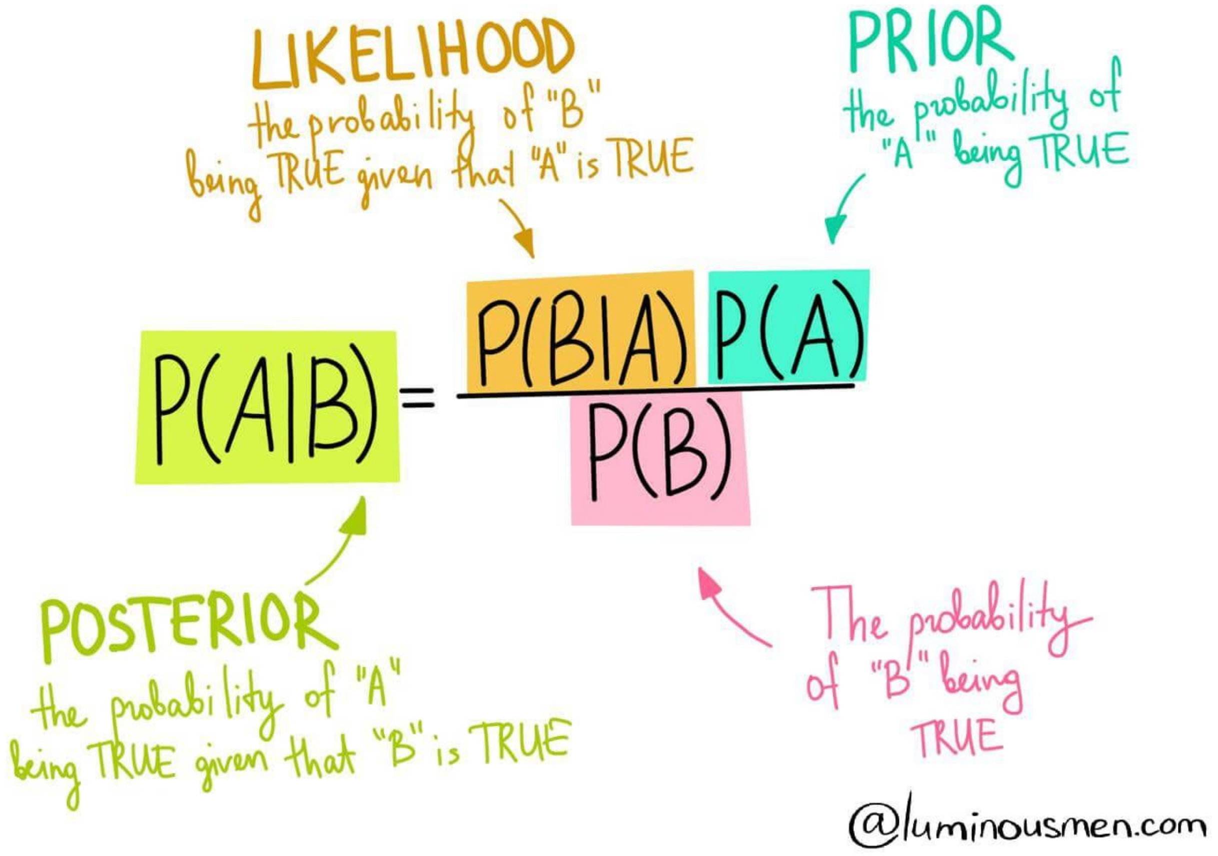
The mean of the Gaussian is the product of the parameters, β and the inputs, X, and the standard deviation is σ.

In Bayesian Models, not only is the response assumed to be sampled from a distribution, **but so are the parameters**. The objective is to determine the posterior probability distribution for the model parameters given the inputs, X, and outputs, y:


LIKELIHOOD
the probability of "B" being TRUE given that "A" is TRUE
PRIOR
the probability of "A" being TRUE
$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}$
POSTERIOR
the probability of "A" being TRUE given that "B" is TRUE
The probability of "B" being TRUE
@luminousmen.com

https://towardsdatascience.com/bayesian-linear-regression-in-python-using-machine-learning-to-predict-student-grades-part-2-b72059a8ac7e

Bayesian and classic statistics think differently about what our data is, where it comes from, and where the uncertainty lies.


LIKELIHOOD
the probability of "B" being TRUE given that "A" is TRUE
PRIOR
the probability of "A" being TRUE
$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}$
POSTERIOR
the probability of "A" being TRUE given that "B" is TRUE
The probability of "B" being TRUE
@luminousmen.com

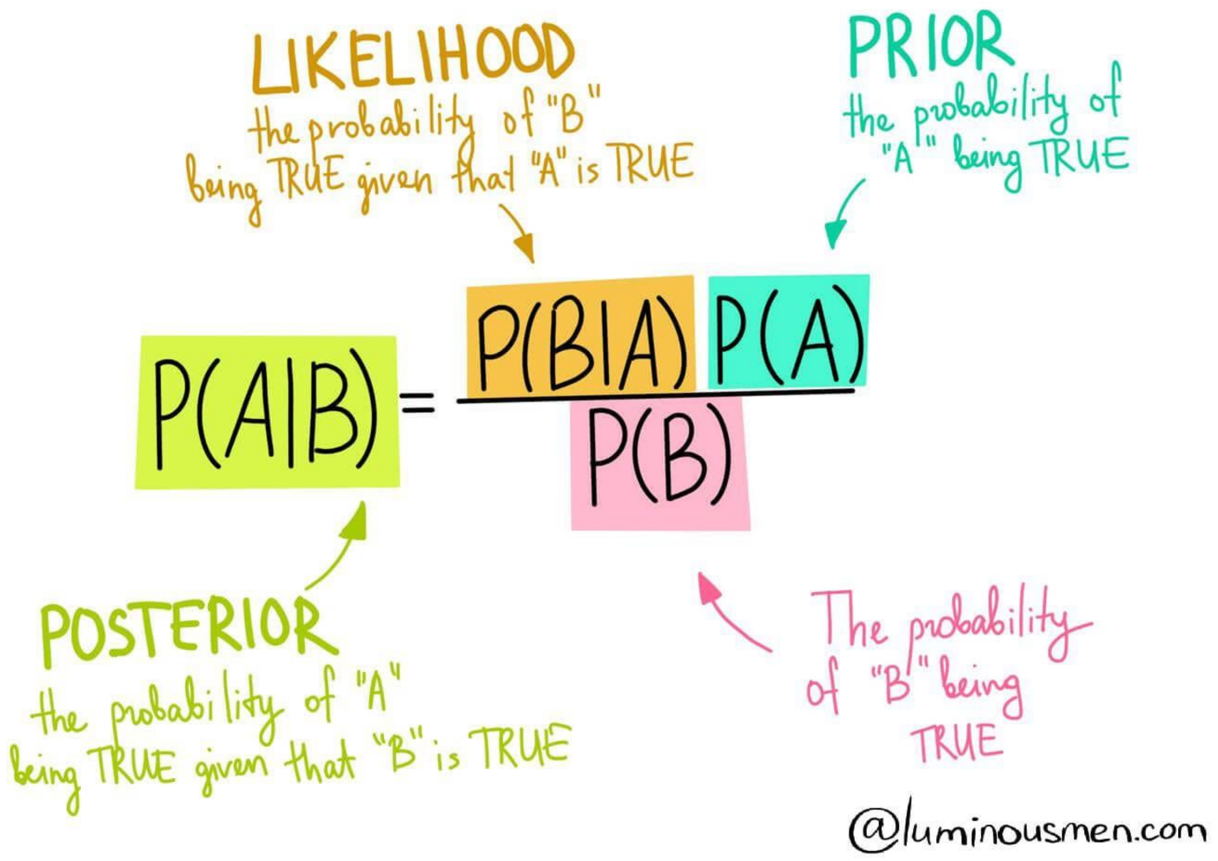
https://towardsdatascience.com/bayesian-linear-regression-in-python-using-machine-learning-to-predict-student-grades-part-2-b72059a8ac7e

Metropolis-Hastings (MH) algorithm, a Markov Chain Monte Carlo (MCMC) approach

1. Initialize:

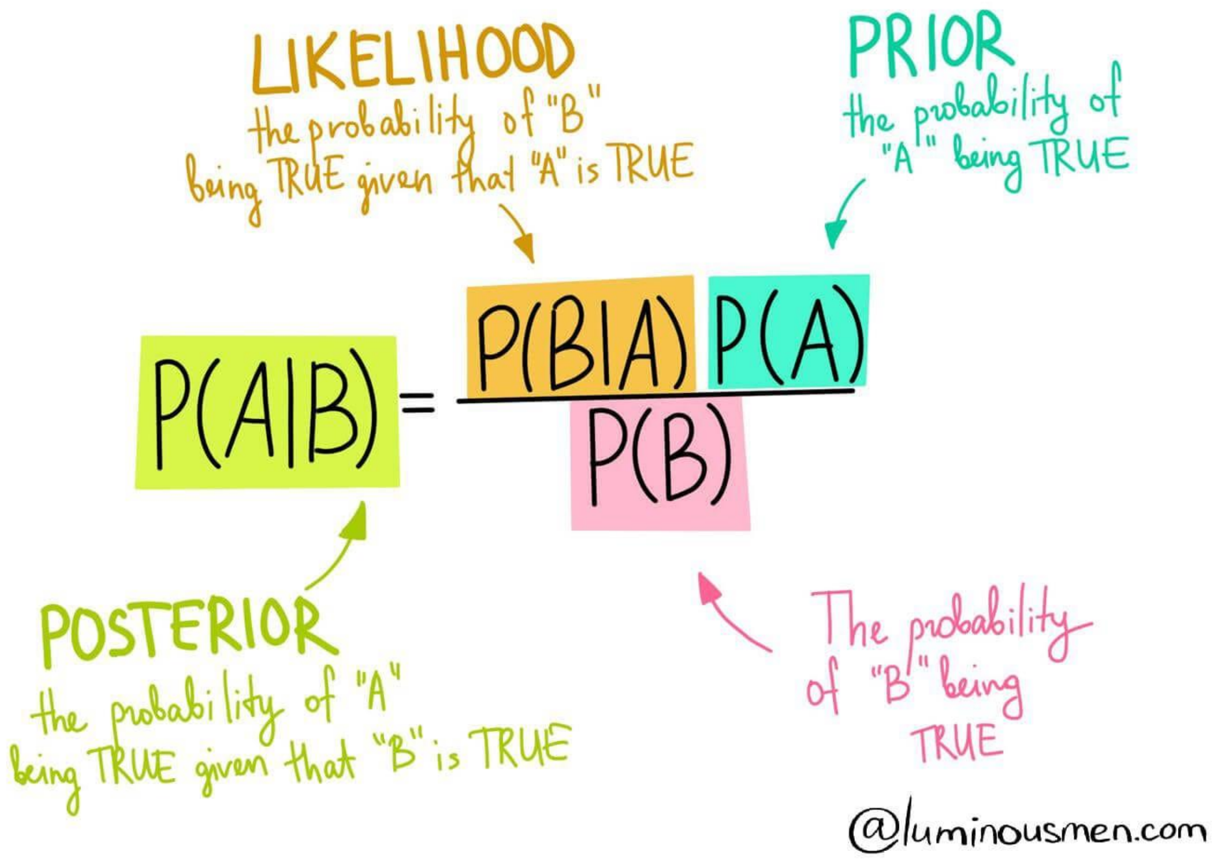
   - Choose a starting point for the regression parameters (beta)

   - Set up storage for MC samples

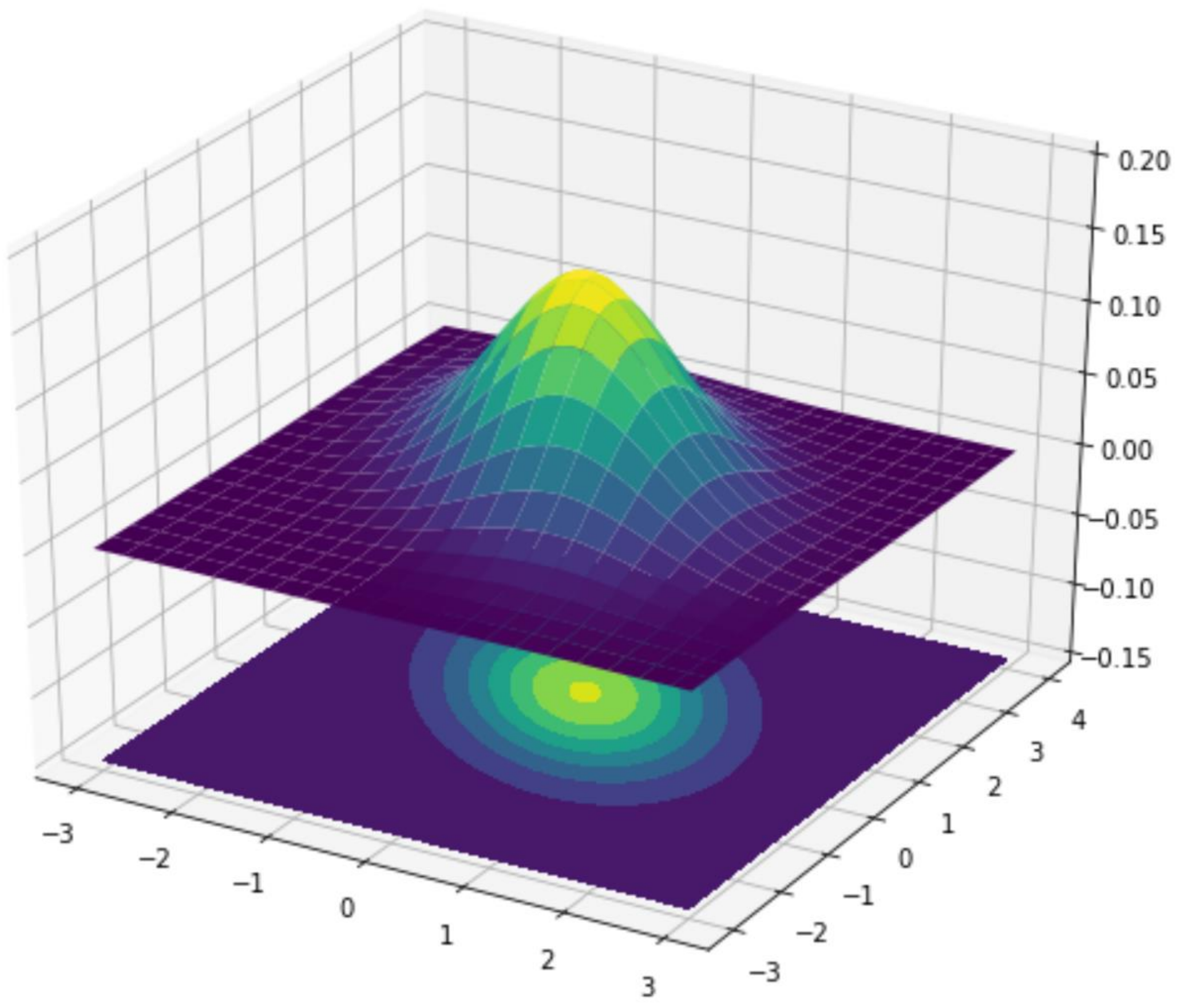

LIKELIHOOD
the probability of "B" being TRUE given that "A" is TRUE
PRIOR
the probability of "A" being TRUE
P(A|B) = P(B|A) P(A) / P(B)
POSTERIOR
the probability of "A" being TRUE given that "B" is TRUE
The probability of "B" being TRUE
@luminousmen.com

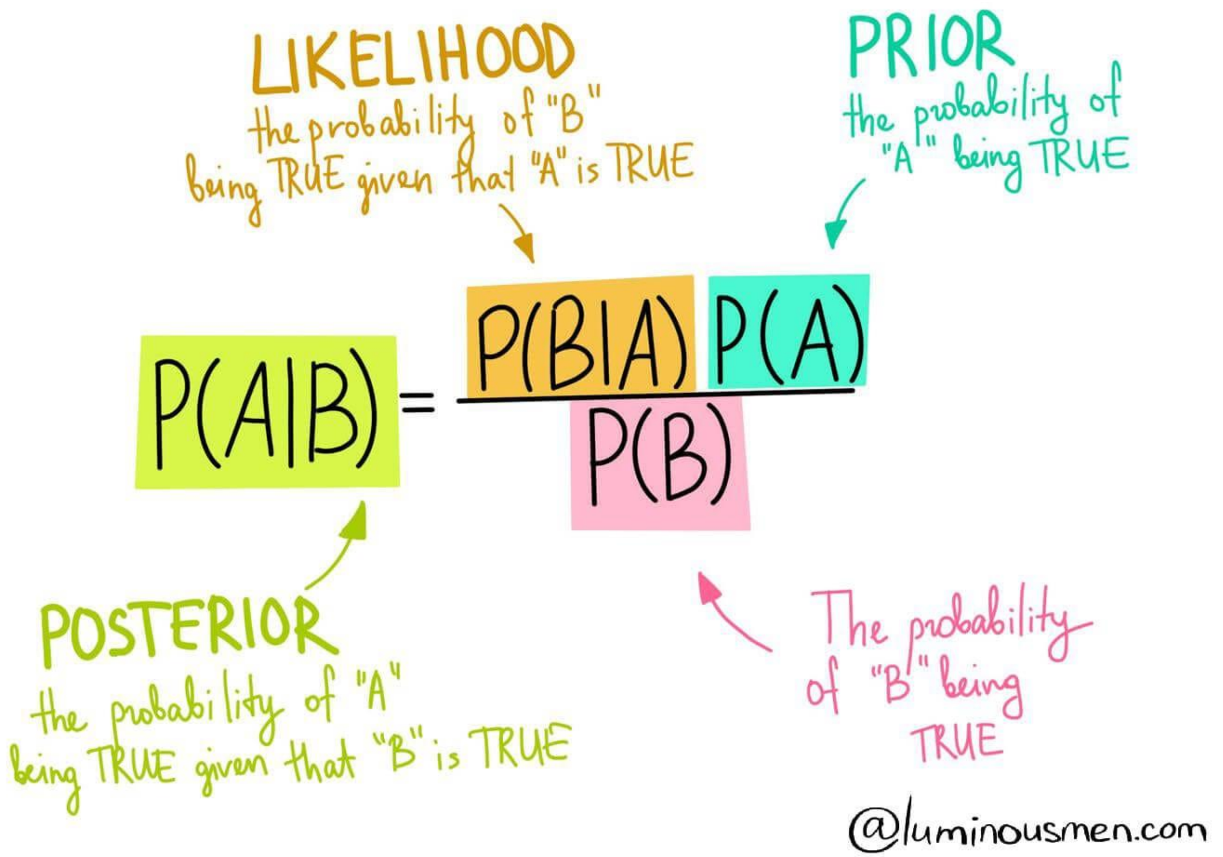
https://towardsdatascience.com/bayesian-linear-regression-in-python-using-machine-learning-to-predict-student-grades-part-2-b72059a8ac7e

```
2. For each iteration i = 1 to N:
   a) Propose a new set of parameters (beta_new) from some proposal distribution (e.g., small Gaussian perturbation around current beta)
   b) Calculate the likelihood of the observed data given beta_new
      likelihood_new = P(y | X, beta_new)
   c) Calculate the likelihood of the observed data given the current beta
      likelihood_current = P(y | X, beta)
   d) Calculate the prior probabilities for beta_new and current beta
      prior_new = P(beta_new)
      prior_current = P(beta)
   e) Calculate the acceptance ratio:
      acceptance_ratio = (likelihood_new * prior_new) / (likelihood_current * prior_current)
   f) Decide whether to accept beta_new:
      - Draw a random number u from a uniform distribution between 0 and 1
      - If u < acceptance_ratio:
         beta = beta_new
   g) Store the sample:
      MCMC_samples[i] = beta
```

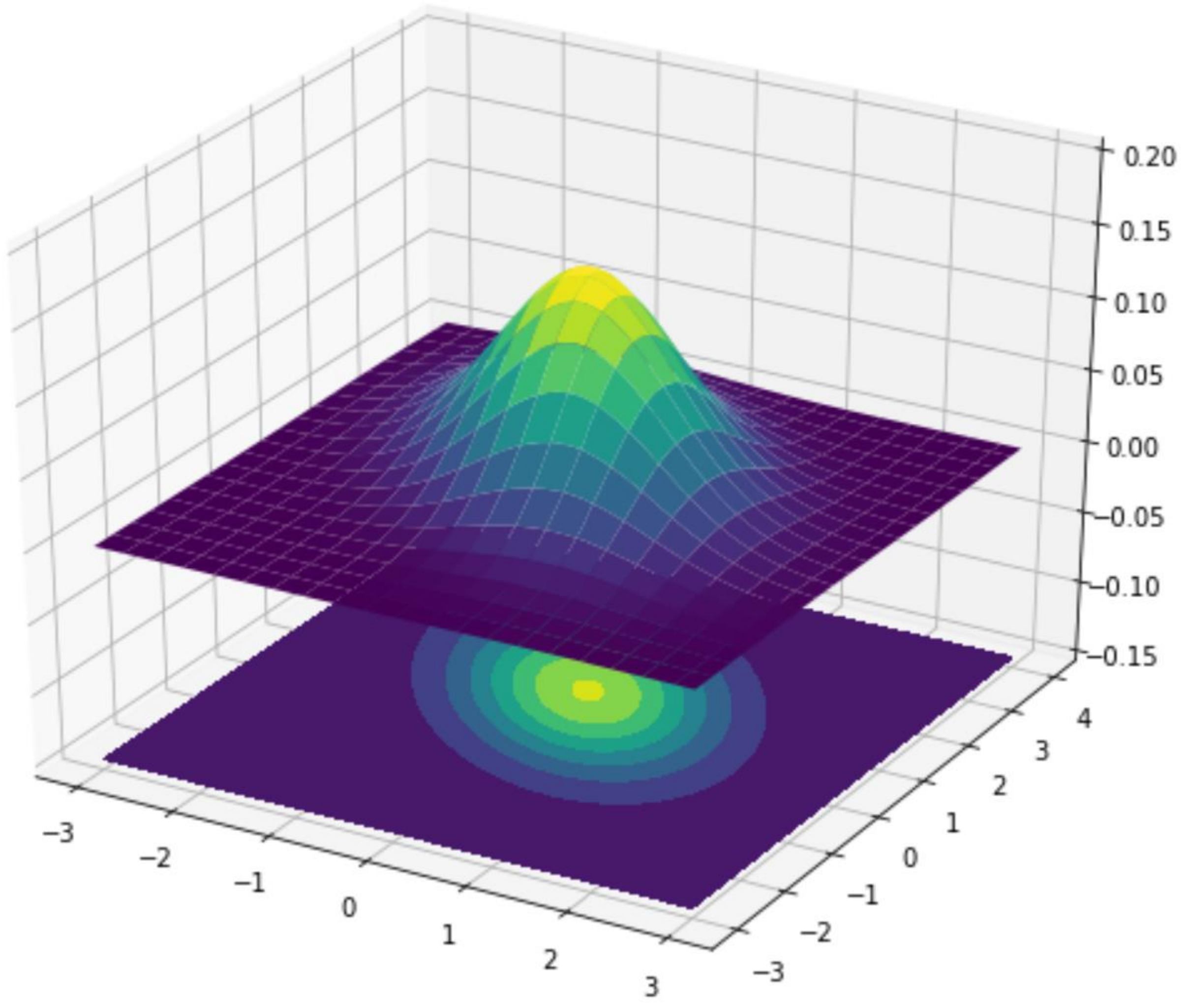

LIKELIHOOD
the probability of "B" being TRUE given that "A" is TRUE
PRIOR
the probability of "A" being TRUE
$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}$
POSTERIOR
the probability of "A" being TRUE given that "B" is TRUE
The probability of "B" being TRUE
@luminousmen.com

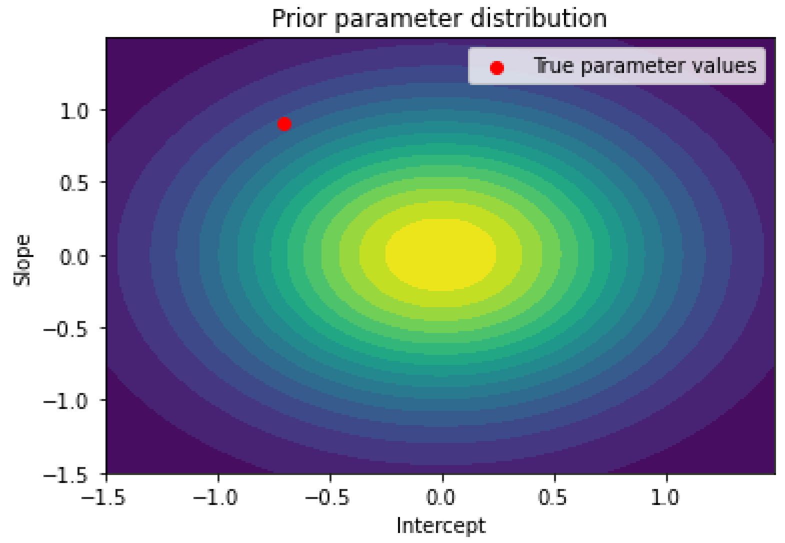
https://towardsdatascience.com/bayesian-linear-regression-in-python-using-machine-learning-to-predict-student-grades-part-2-b72059a8ac7e



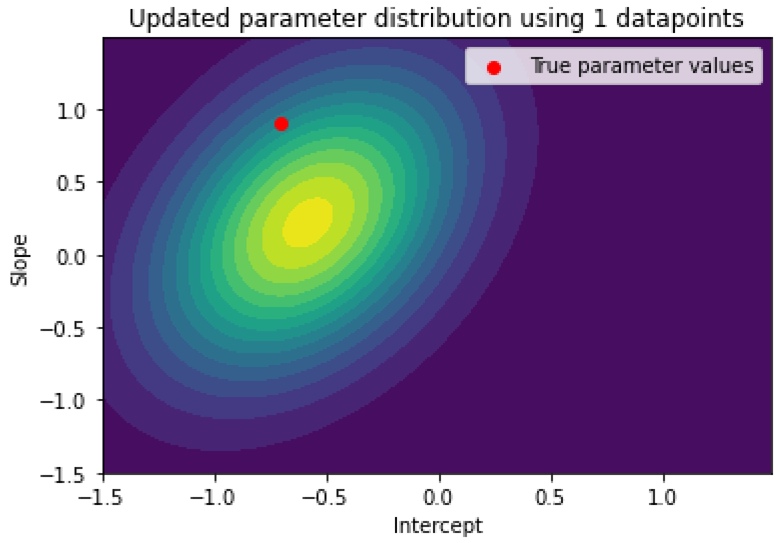

LIKELIHOOD
the probability of "B" being TRUE given that "A" is TRUE
PRIOR
the probability of "A" being TRUE
P(A|B) = P(B|A) P(A) / P(B)
POSTERIOR
the probability of "A" being TRUE given that "B" is TRUE
The probability of "B" being TRUE
@luminousmen.com

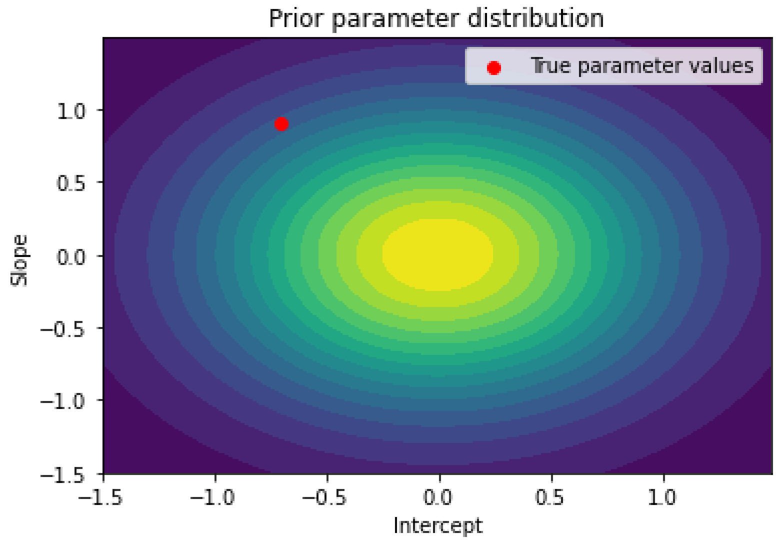
https://github.com/UmaMaquinaDeOndas/machine
_learning_basics/blob/master/bayesian_linear_regre
sion.ipynb



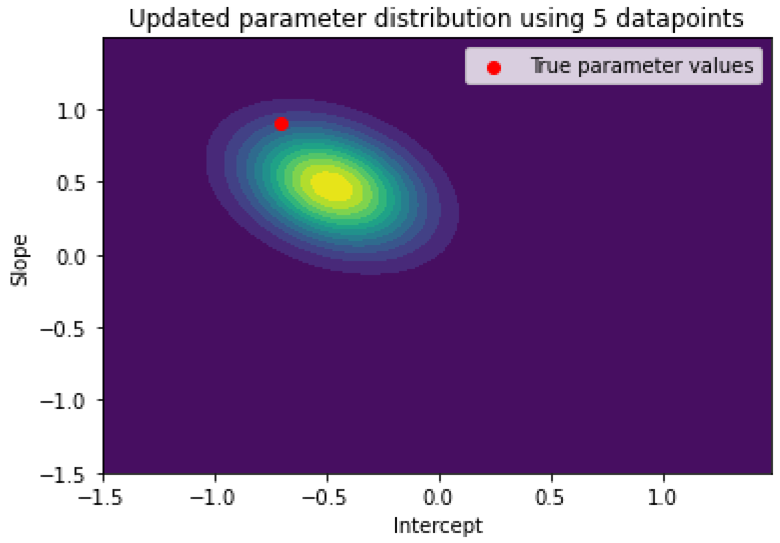


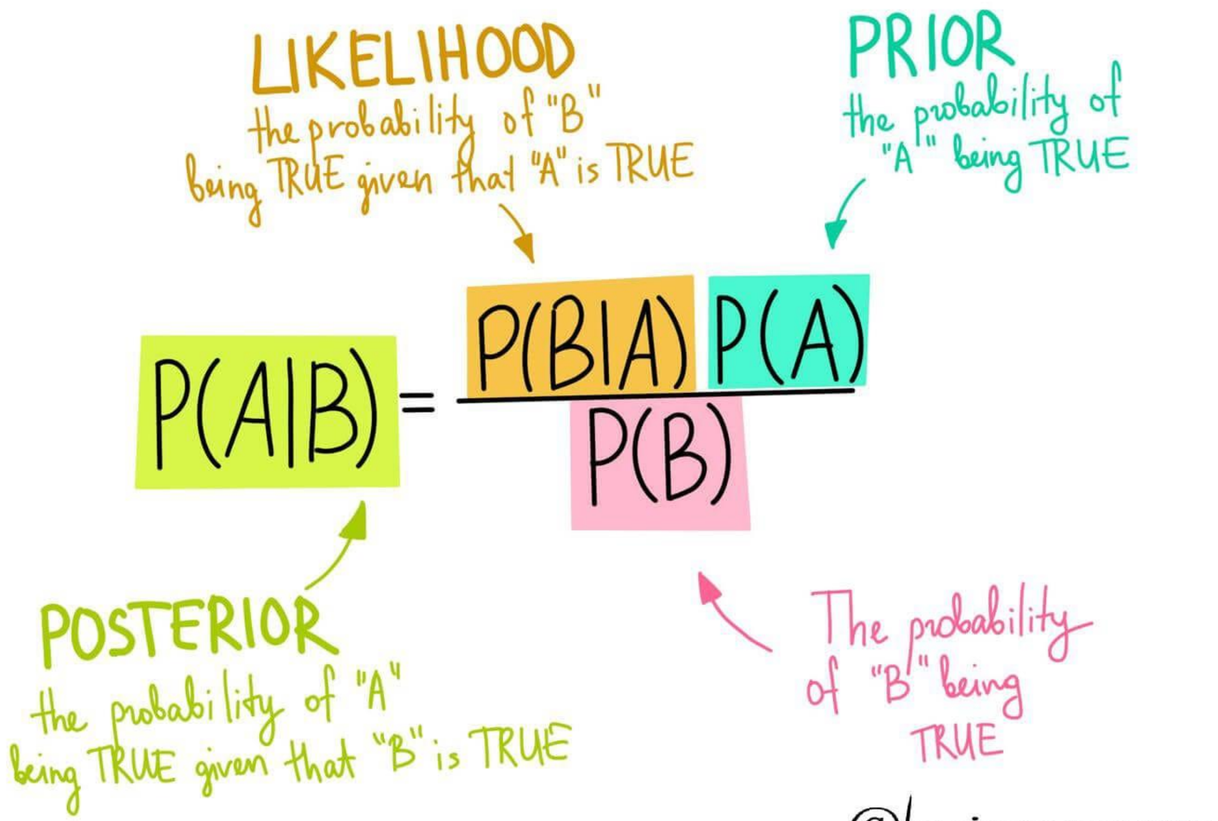
https://github.com/UmaMaquinaDeOndas/machine
_learning_basics/blob/master/bayesian_linear_regre
sion.ipynb



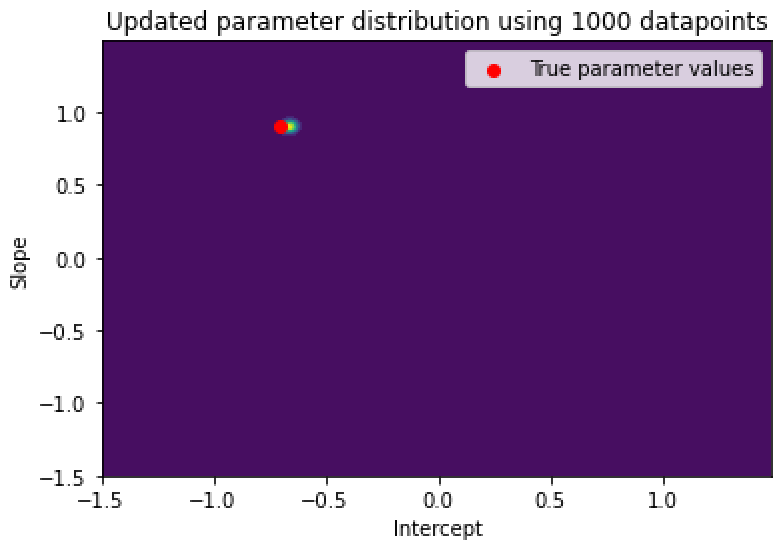


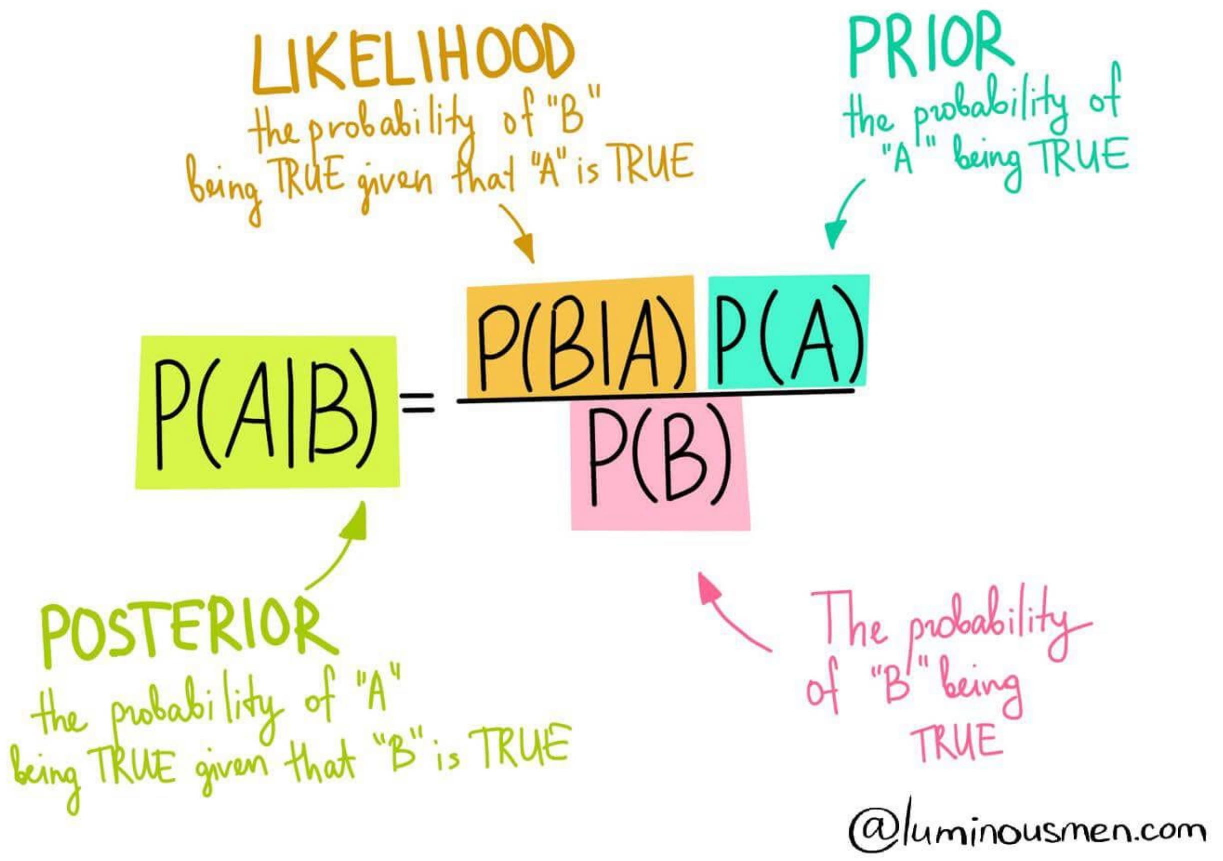
https://github.com/UmaMaquinaDeOndas/machine
_learning_basics/blob/master/bayesian_linear_regre
sion.ipynb



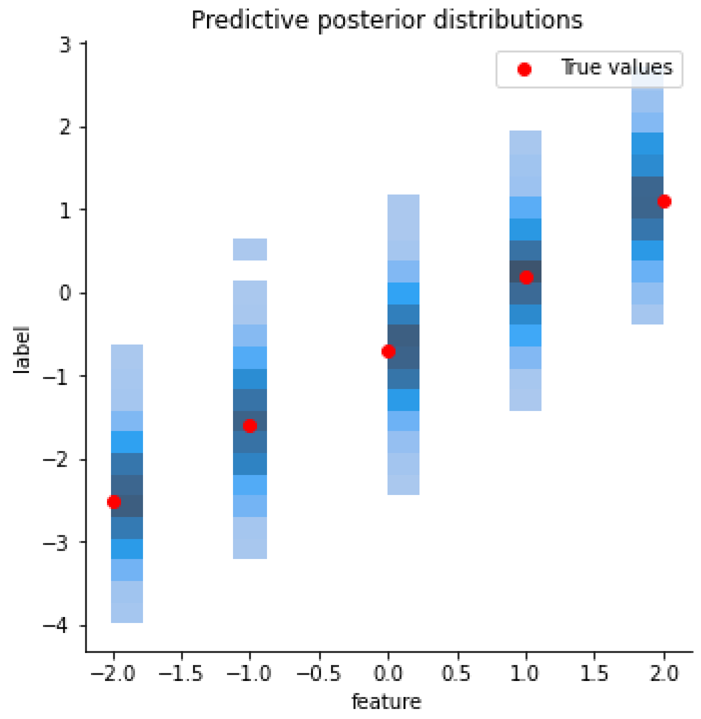

LIKELIHOOD
the probability of "B" being TRUE given that "A" is TRUE
PRIOR
the probability of "A" being TRUE
$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}$
POSTERIOR
the probability of "A" being TRUE given that "B" is TRUE
The probability of "B" being TRUE
@luminousmen.com

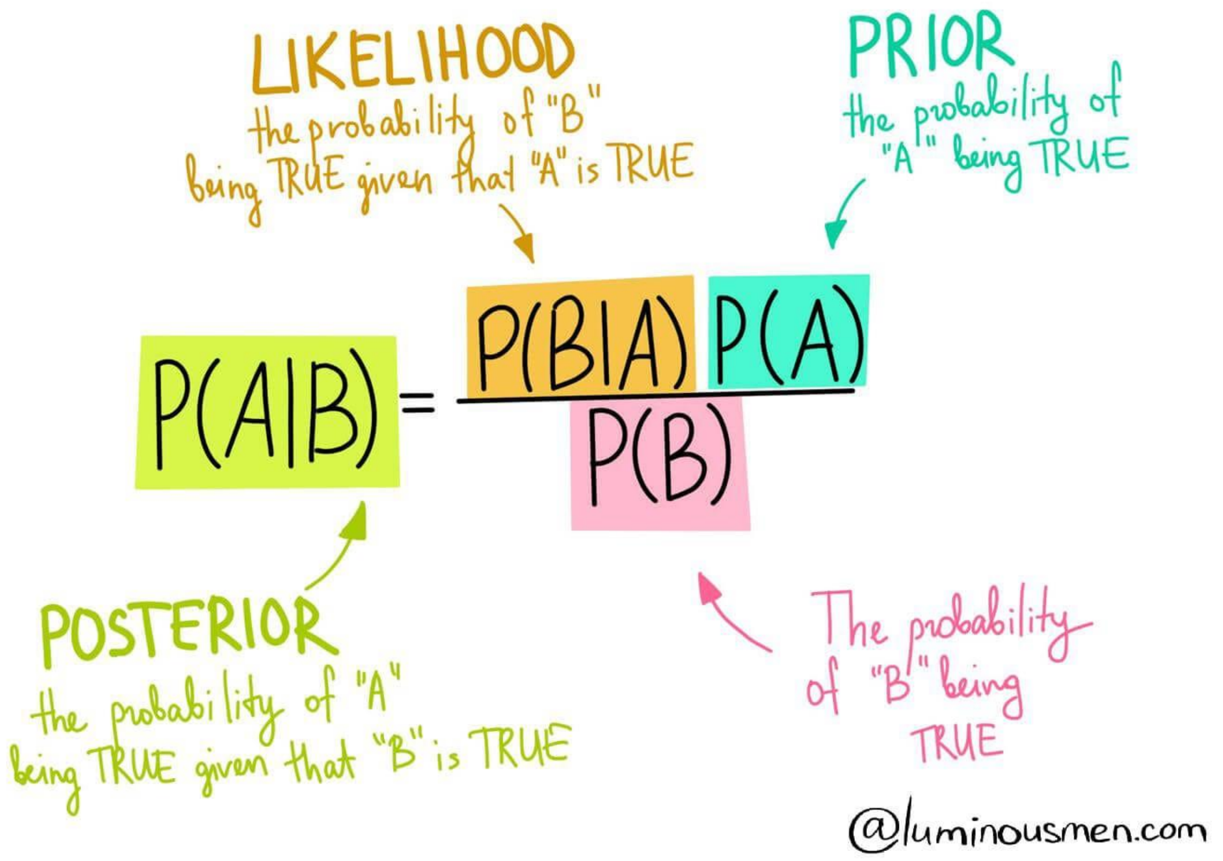
https://github.com/UmaMaquinaDeOndas/machine
_learning_basics/blob/master/bayesian_linear_regre
sion.ipynb




LIKELIHOOD
the probability of "B" being TRUE given that "A" is TRUE
PRIOR
the probability of "A" being TRUE
$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}$
POSTERIOR
the probability of "A" being TRUE given that "B" is TRUE
The probability of "B" being TRUE
@luminousmen.com

https://github.com/UmaMaquinaDeOndas/machine
_learning_basics/blob/master/bayesian_linear_regre
sion.ipynb



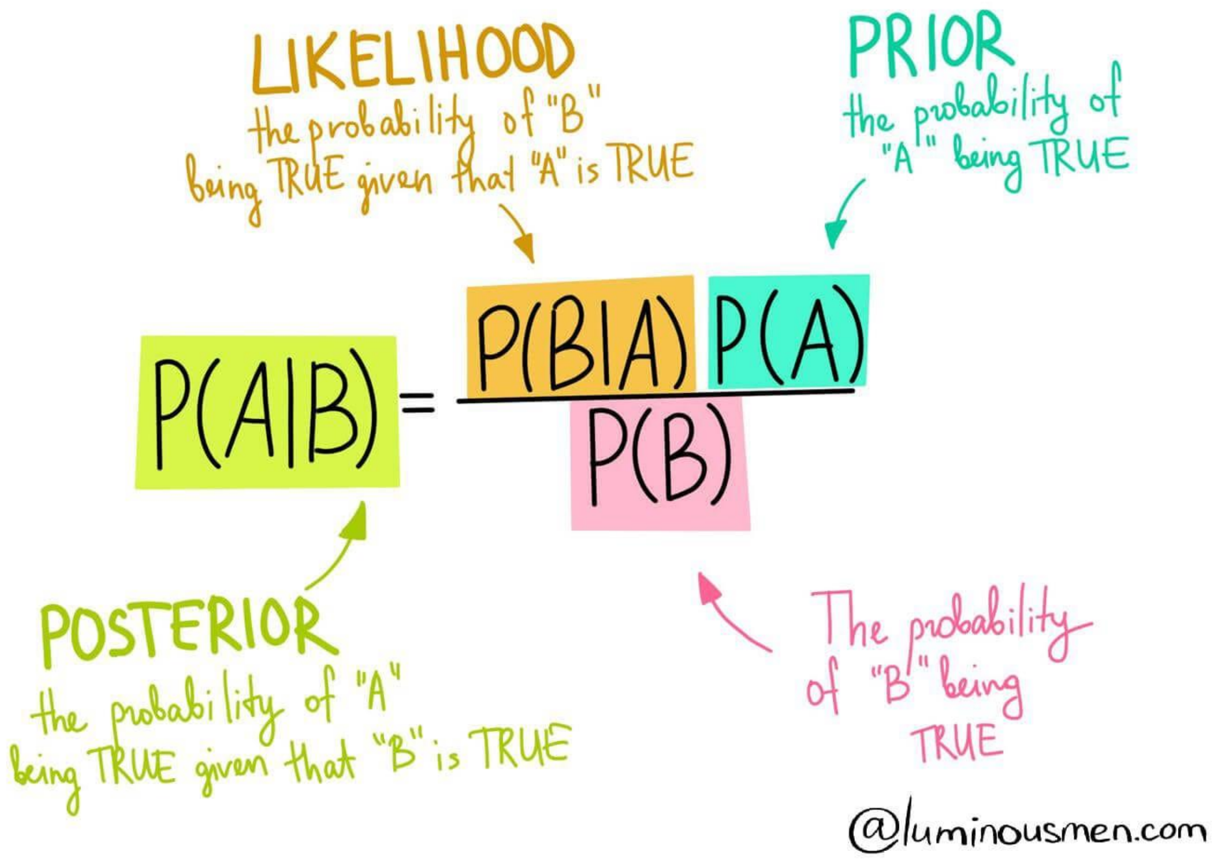


LIKELIHOOD
the probability of "B" being TRUE given that "A" is TRUE
PRIOR
the probability of "A" being TRUE
P(A|B) = P(B|A) P(A) / P(B)
POSTERIOR
the probability of "A" being TRUE given that "B" is TRUE
The probability of "B" being TRUE
@luminousmen.com

https://github.com/UmaMaquinaDeOndas/machine
_learning_basics/blob/master/bayesian_linear_regre
sion.ipynb

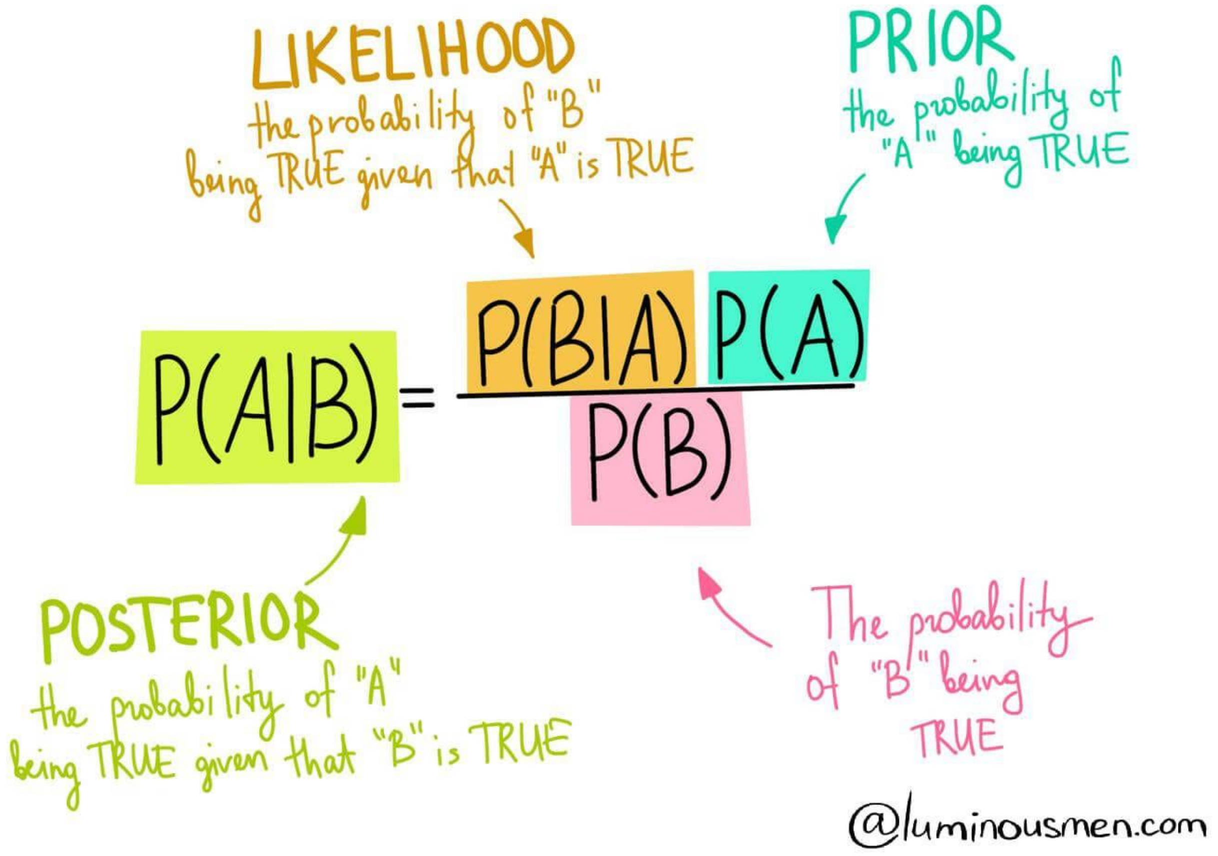
scikit-learn's BayesianRidge – Simple normal distribution around parameters and analytical approximation of posterior.

PyMC – Full distribution specification and PyMC uses Markov Chain Monte Carlo (MCMC) or variational inference for sampling the posterior.


LIKELIHOOD
the probability of "B" being TRUE given that "A" is TRUE
PRIOR
the probability of "A" being TRUE
P(A|B) = P(B|A) P(A) / P(B)
POSTERIOR
the probability of "A" being TRUE given that "B" is TRUE
The probability of "B" being TRUE
@luminousmen.com

https://towardsdatascience.com/bayesian-linear-regression-in-python-using-machine-learning-to-predict-student-grades-part-2-b72059a8ac7e

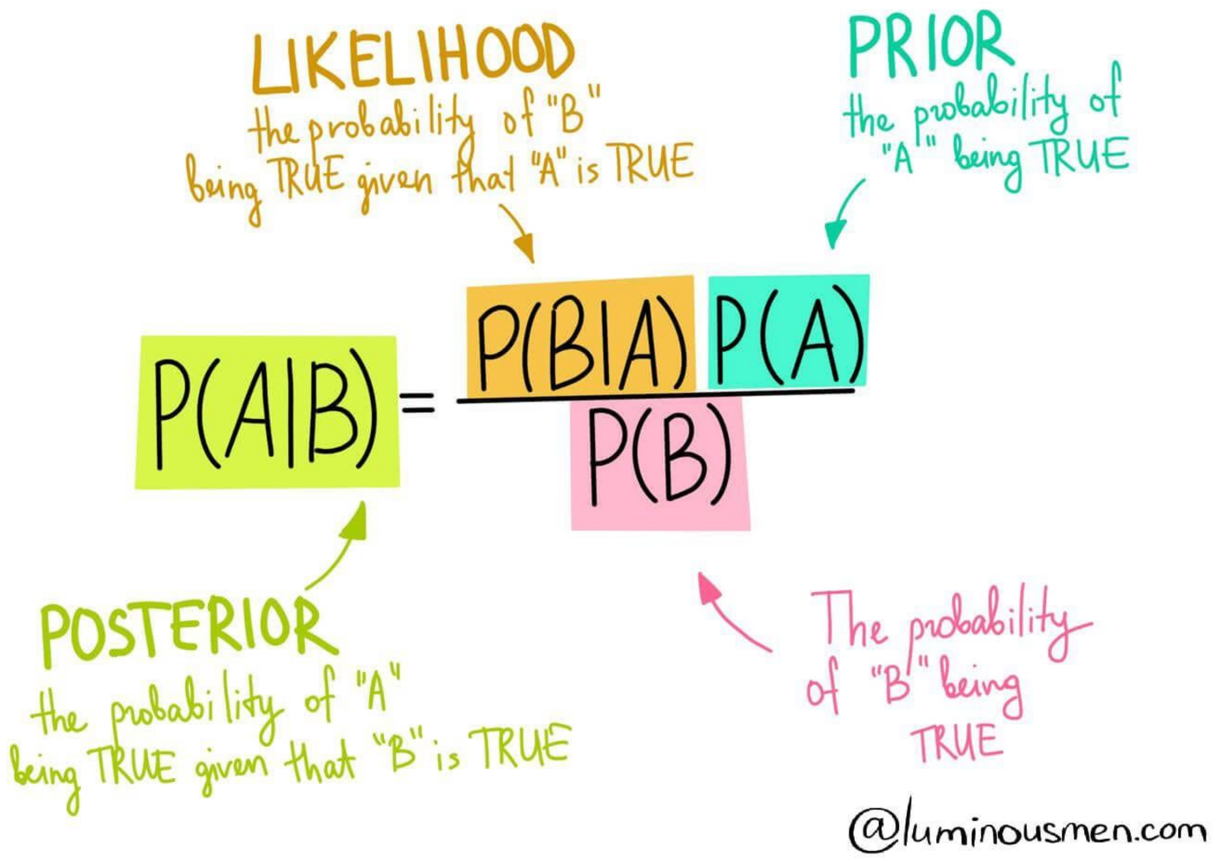
It all comes down to priors.


LIKELIHOOD
the probability of "B" being TRUE given that "A" is TRUE
PRIOR
the probability of "A" being TRUE
$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}$
POSTERIOR
the probability of "A" being TRUE given that "B" is TRUE
The probability of "B" being TRUE
@luminousmen.com

https://towardsdatascience.com/bayesian-linear-regression-in-python-using-machine-learning-to-predict-student-grades-part-2-b72059a8ac7e

Linear models are versatile and mathematically tractable but often too rigid to capture intricate relationships in data.

Splines serve as a natural extension, providing a piecewise-defined polynomial approach to modeling data, while maintaining computational efficiency.

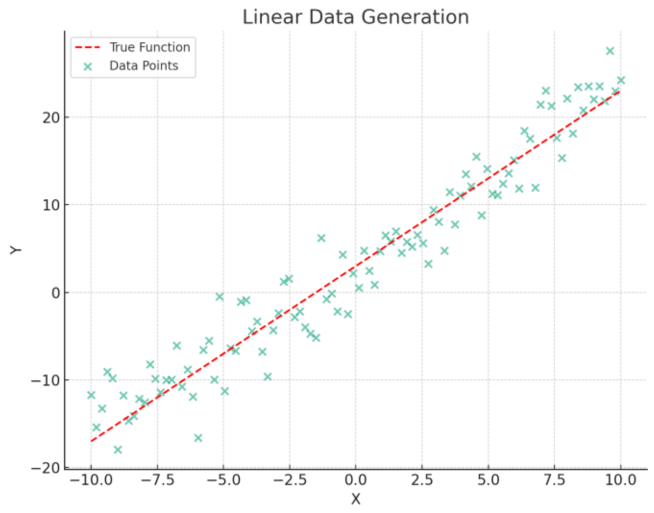

LIKELIHOOD
the probability of "B" being TRUE given that "A" is TRUE
PRIOR
the probability of "A" being TRUE
P(A|B) = P(B|A) P(A) / P(B)
POSTERIOR
the probability of "A" being TRUE given that "B" is TRUE
The probability of "B" being TRUE
@luminousmen.com

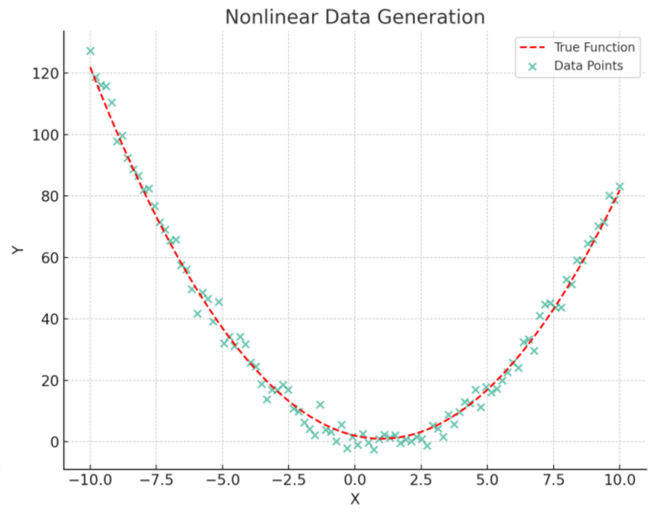
https://towardsdatascience.com/bayesian-linear-regression-in-python-using-machine-learning-to-predict-student-grades-part-2-b72059a8ac7e

- In a polynomial regression, the model includes terms that are powers of the predictor variable(s). For a single predictor x, the model is:

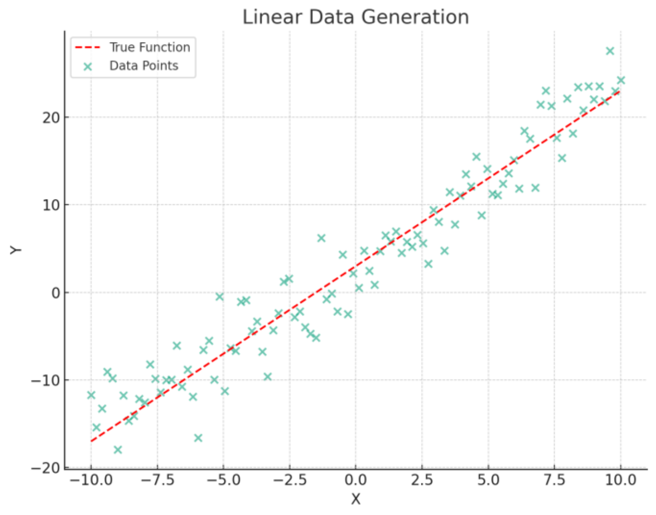
$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \cdots + \beta_d x^d + \epsilon$$

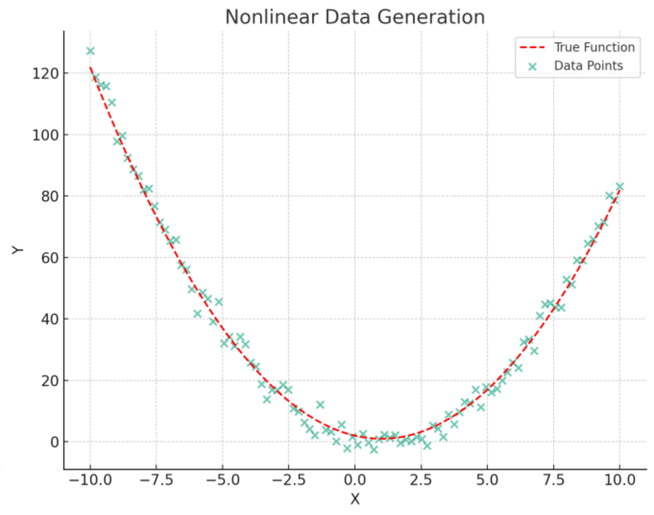
- For multiple predictors, the model can include interactions and higher-order terms for each predictor.





- **Overfitting:** Higher-degree polynomials can fit the data too closely, capturing noise rather than the underlying pattern. Regularization or cross-validation can help mitigate this.
- **Extrapolation:** Polynomial models can behave unpredictably outside the range of the data.
- **Collinearity:** Higher-degree terms (e.g., $x2,x3$x 2 ,x 3) can be highly correlated with the lower-degree terms (xx), which can lead to unstable estimates of the coefficients.





- **Overfitting:** Higher-degree polynomials can fit the data too closely, capturing noise rather than the underlying pattern. Regularization or cross-validation can help mitigate this.
- **Extrapolation:** Polynomial models can behave unpredictably outside the range of the data.
- **Collinearity:** Higher-degree terms (e.g., x2,x3x 2 ,x 3) can be highly correlated with the lower-degree terms (xx), which can lead to unstable estimates of the coefficients.

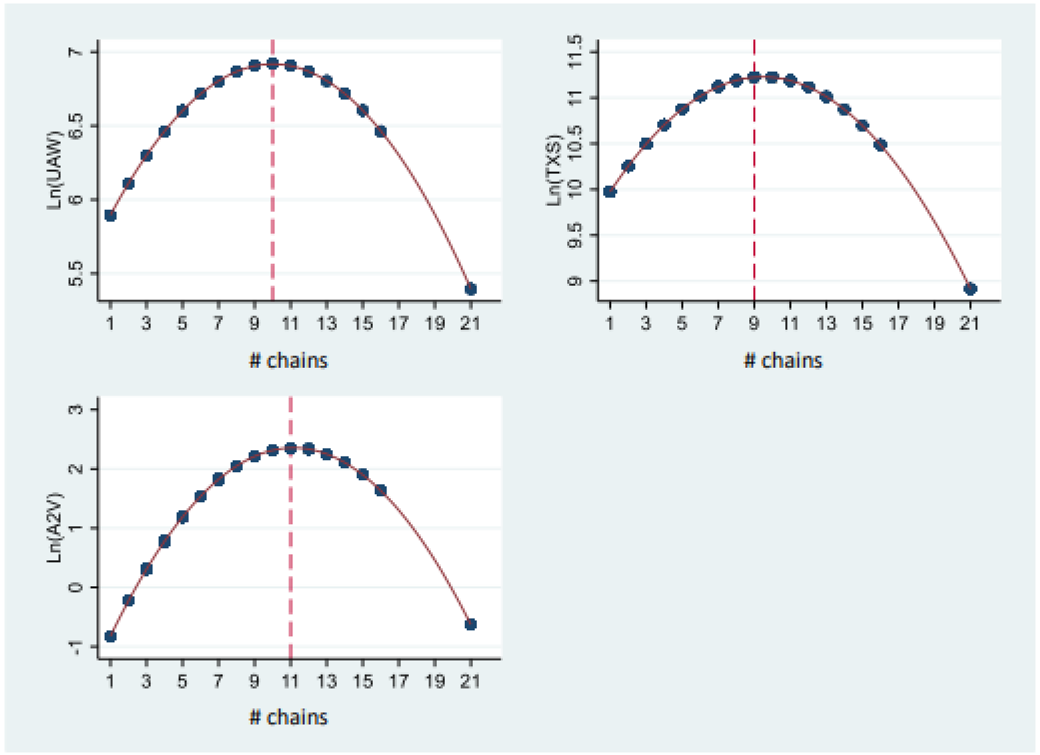
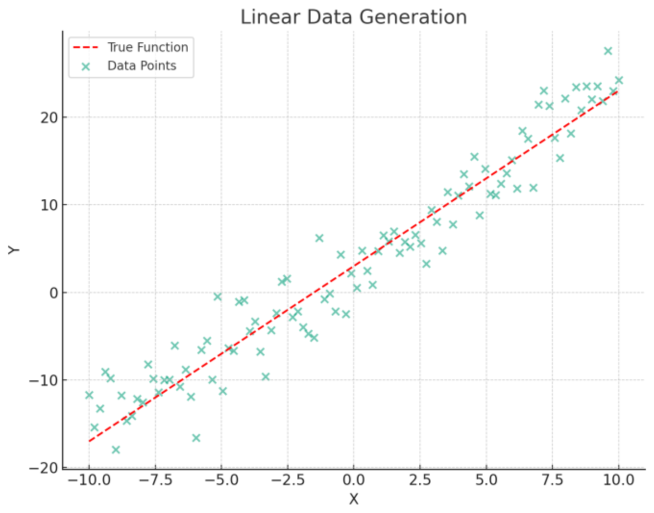
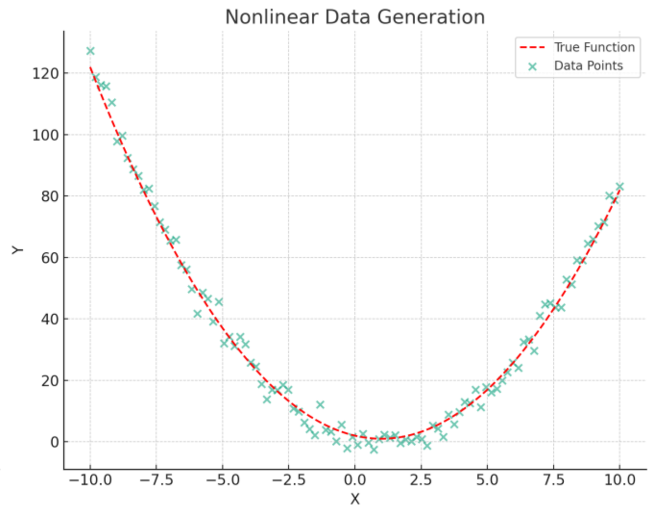


Figure 10: Fitted plots of the quadratic regression results

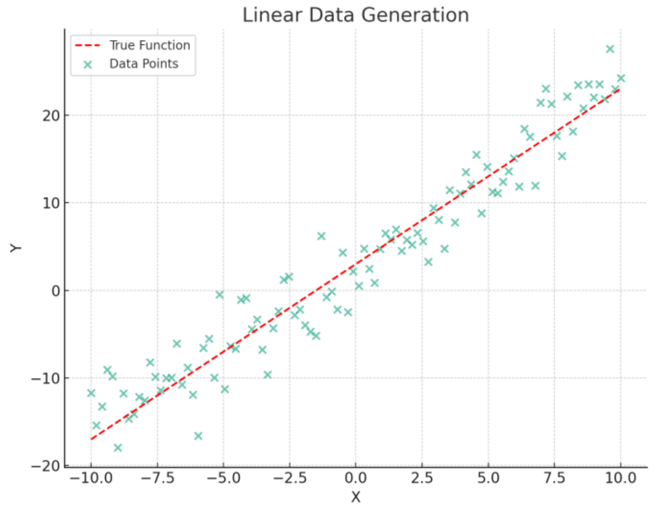
Pereira, Joana and Tavalaei, M.Mahdi and Herrera, Pedro and Rebo, Boris, Cross-Chain Dapps: How Many is Too Many Chains? (August 2, 2023). Available at SSRN: https://ssrn.com/abstract=4528889 or http://dx.doi.org/10.2139/ssrn.4528889

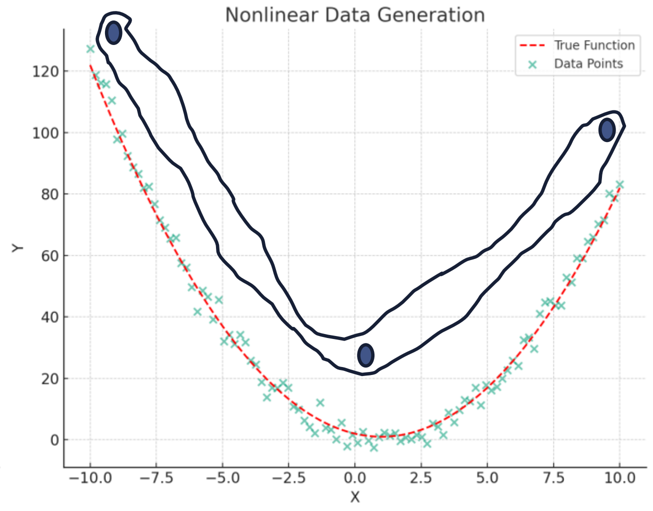
Splines serve as a natural extension, providing a piecewise-defined polynomial approach to modeling data, while maintaining computational efficiency.





Imagine a rubber band stretched across a series of pegs (knots). The rubber band will naturally form a smooth curve that is tightly pulled at each knot but flexible in between. This curve embodies the essential features of a spline.

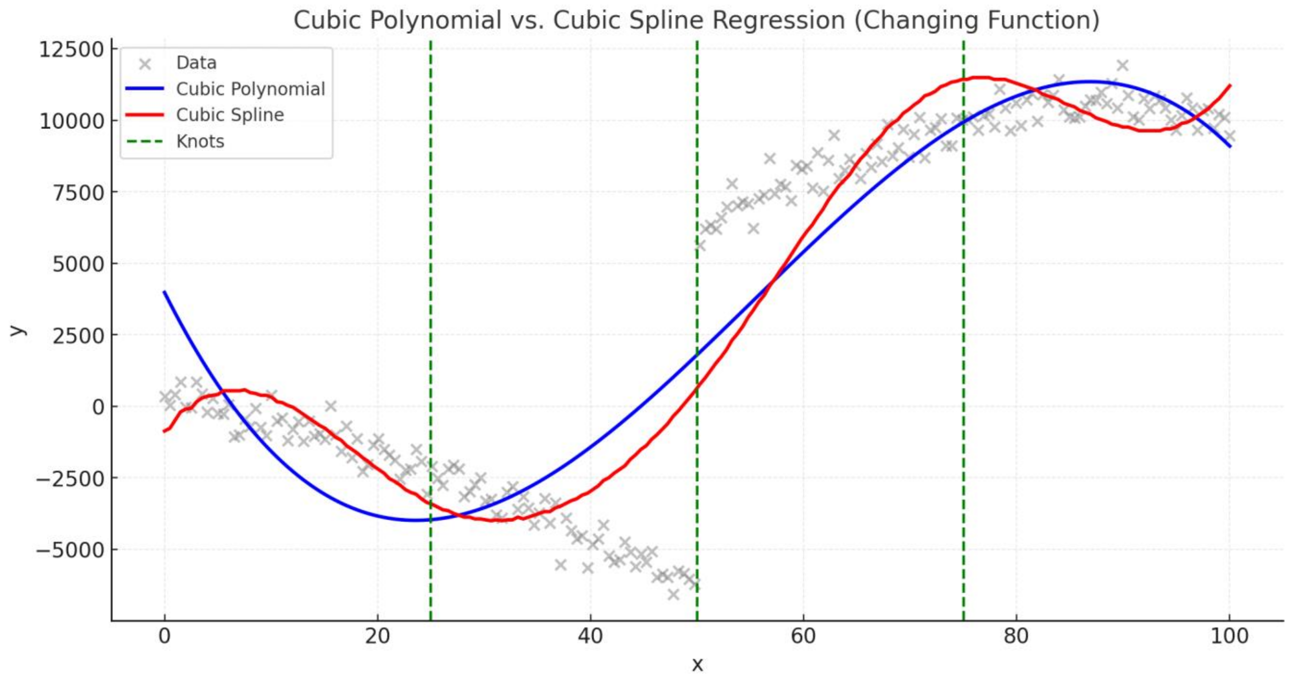




In practice we apply multiple splines, imagining that different splines fit the data better in different places.

$$s(x) = \begin{cases} a_1 + b_1x + c_1x^2 + d_1x^3 & \text{for } x < t_1 \\ a_2 + b_2x + c_2x^2 + d_2x^3 & \text{for } t_1 \leq x < t_2 \\ \vdots & \\ a_k + b_kx + c_kx^2 + d_kx^3 & \text{for } x \geq t_{k-1} \end{cases}$$

In practice we apply multiple splines, imagining that different splines fit the data better in different places.



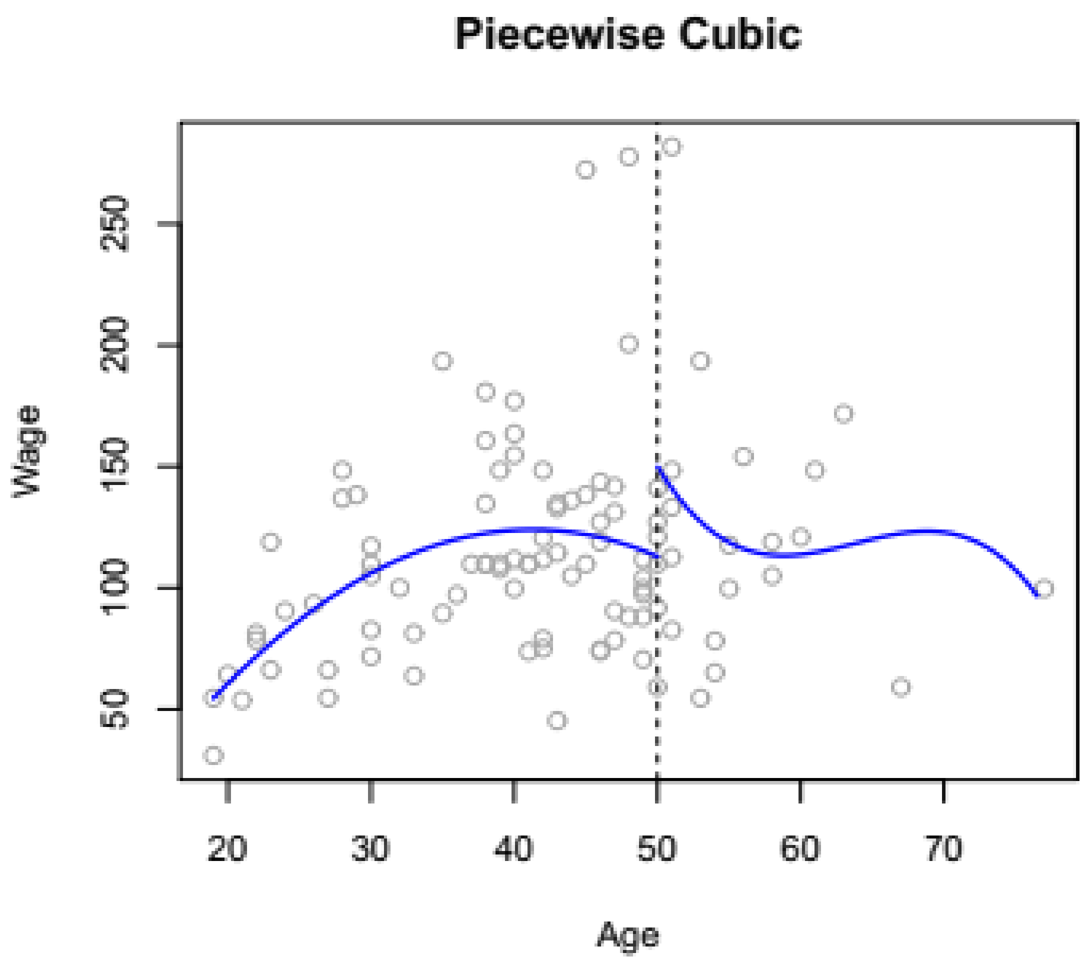
a knot is a specific value of the independent variable where the form of the polynomial function changes.

Knots **partition the domain of the independent variable into intervals.**

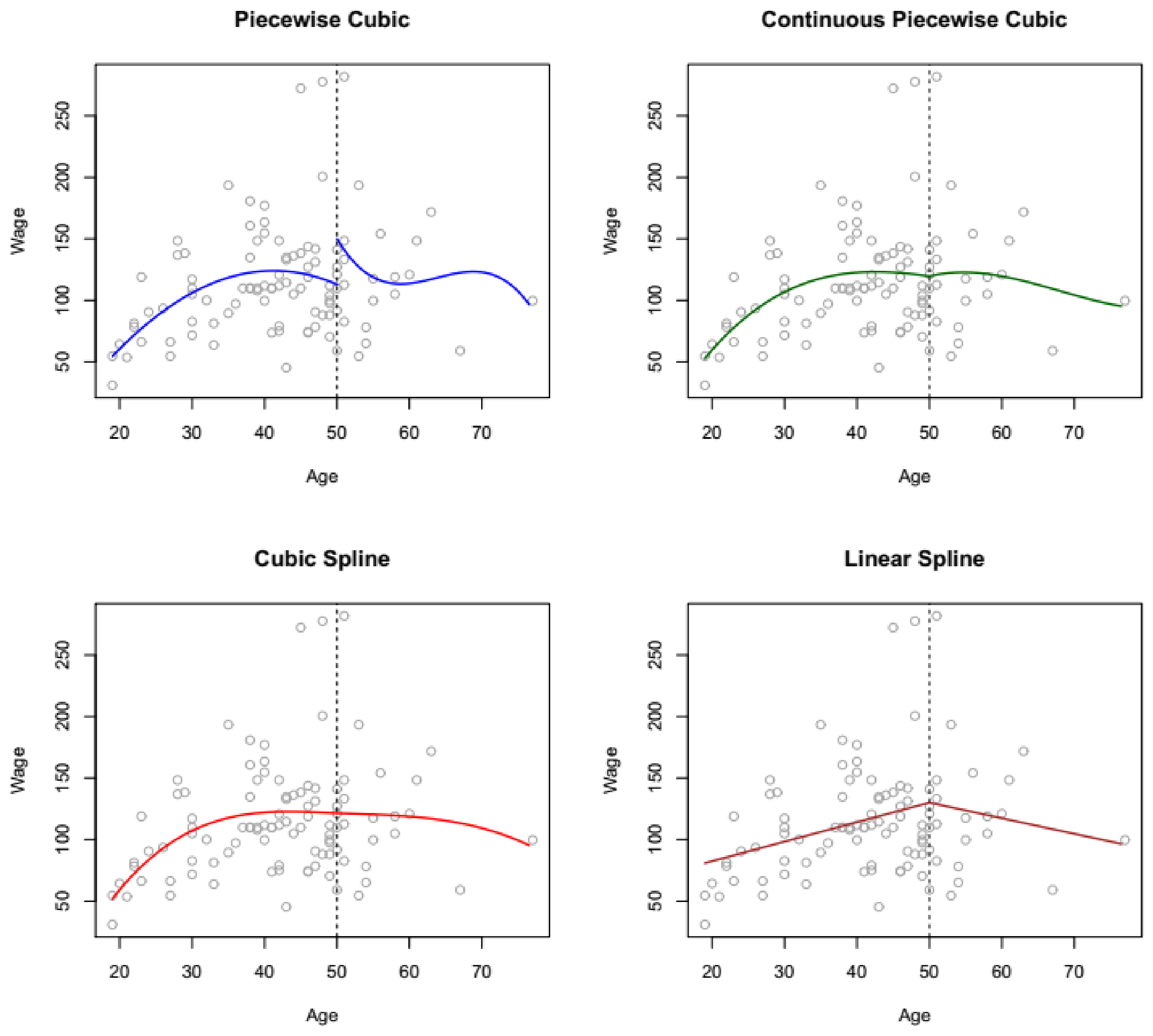
Within each interval, a different polynomial function describes the relationship between the dependent and independent variables.

$$s(x) = \begin{cases} a_1 + b_1 x + c_1 x^2 + d_1 x^3 & \text{for } x < t_1 \\ a_2 + b_2 x + c_2 x^2 + d_2 x^3 & \text{for } t_1 \leq x < t_2 \\ \vdots \\ a_k + b_k x + c_k x^2 + d_k x^3 & \text{for } x \geq t_{k-1} \end{cases}$$

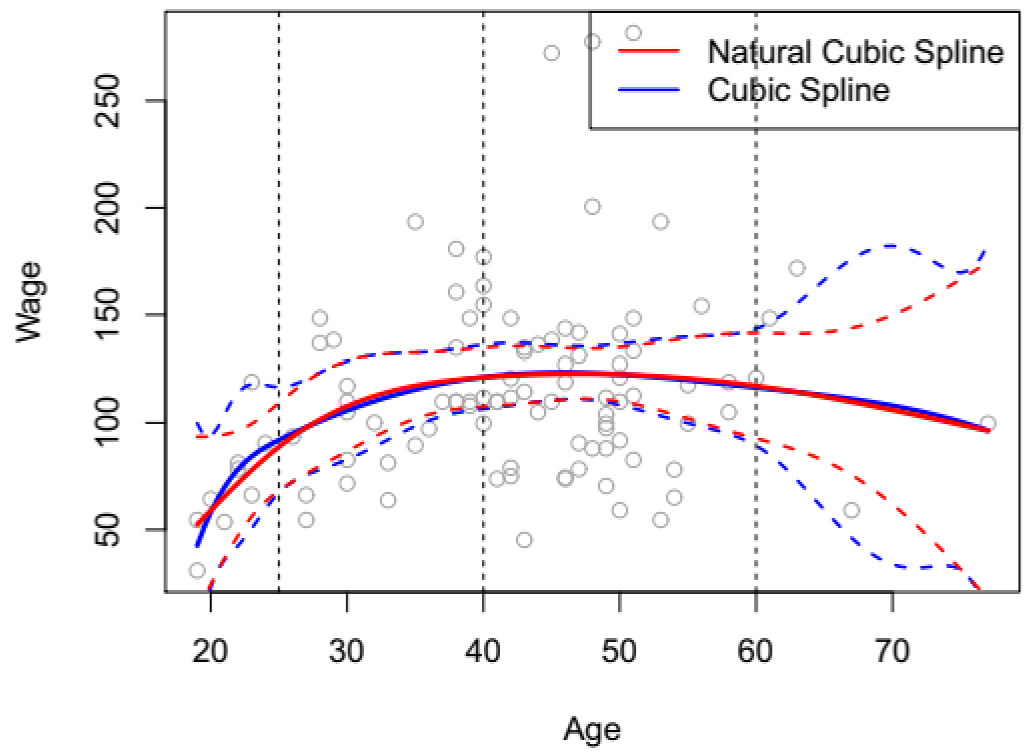
It's worth mentioning that to ensure a smooth curve, additional constraints are usually placed at the knots.



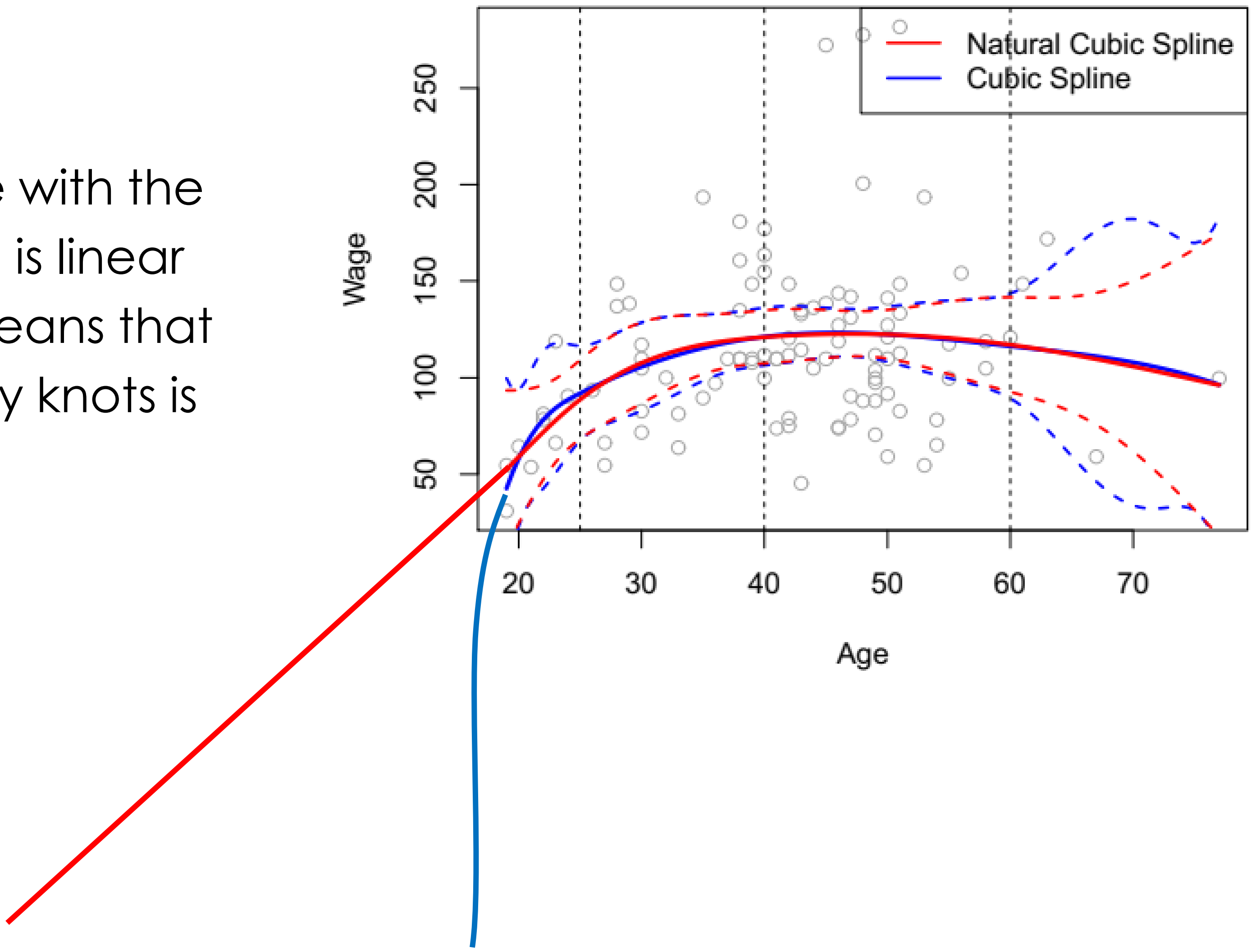
For cubic splines, the first and second derivatives are often set to be continuous at each knot. This ensures that there are no abrupt changes in the function or its rate of change at these points.



A **natural cubic spline** is a cubic spline with the additional constraint that the function is linear beyond the boundary knots, which means that the second derivative at the boundary knots is zero.

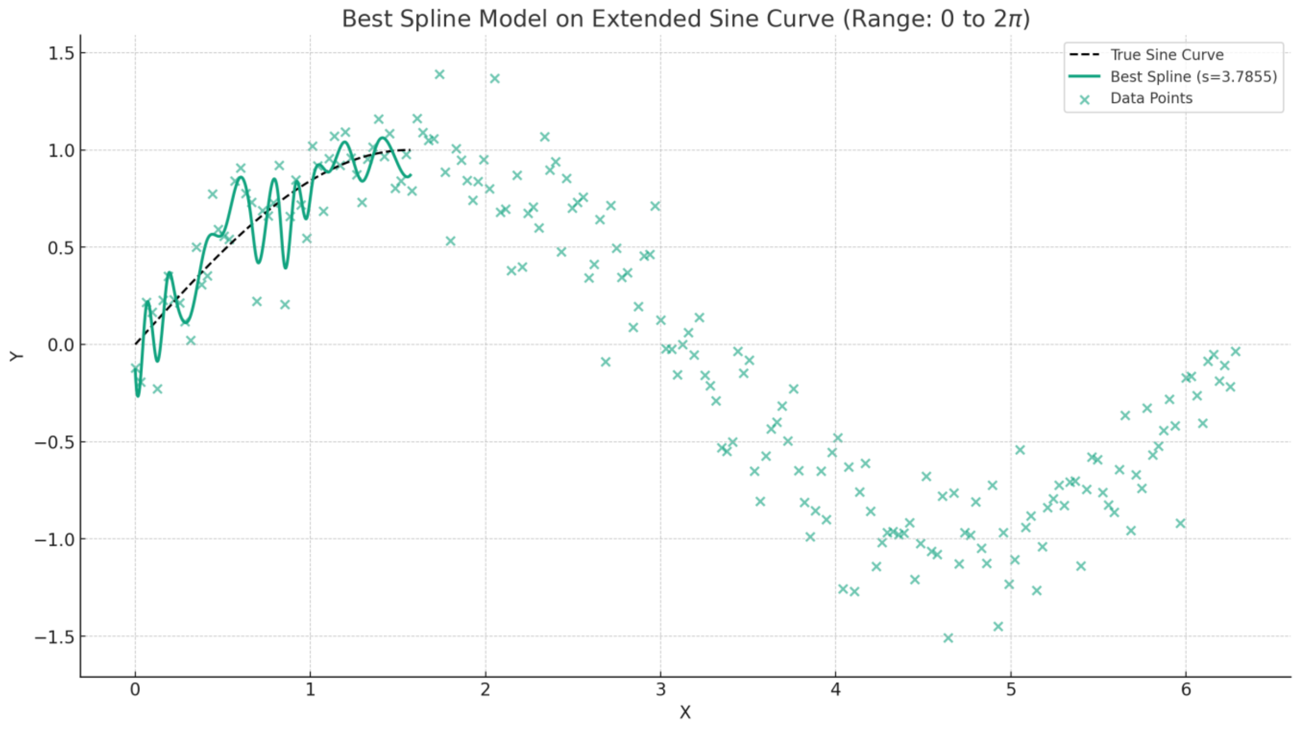


A **natural cubic spline** is a cubic spline with the additional constraint that the function is linear beyond the boundary knots, which means that the second derivative at the boundary knots is zero.



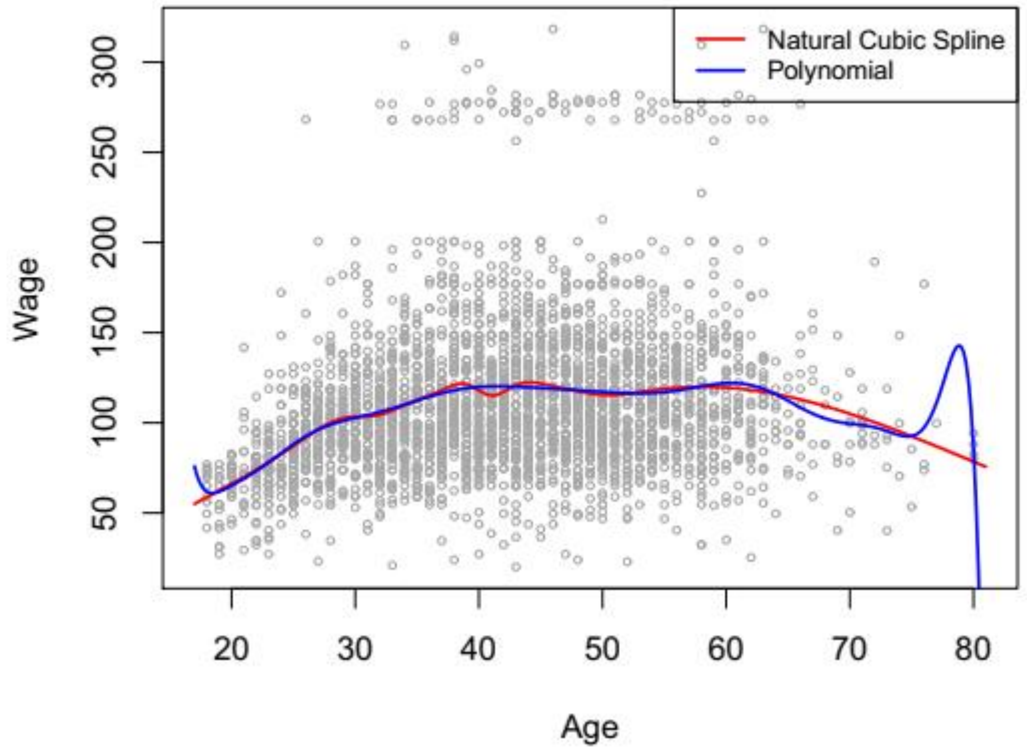
Bias – Variance Trade-off - Knots offer a trade-off between flexibility and overfitting.

- More knots allow for greater flexibility but risk overfitting
- fewer knots may result in a smoother curve that potentially underfits the data.
- Techniques like cross-validation are often employed to select an optimal number of knots.



Bias – Variance Trade-off - Knots offer a trade-off between flexibility and overfitting.

- More knots allow for greater flexibility but risk overfitting
- fewer knots may result in a smoother curve that potentially underfits the data.
- Techniques like cross-validation are often employed to select an optimal number of knots.



Advantages

- Flexibility
- Easy to interpret
- Computational efficiency

Limitations

- Choice of knots
- Potential for overfitting
- Boundary issues

$$s(x) = \begin{cases} a_1 + b_1 x + c_1 x^2 + d_1 x^3 & \text{for } x < t_1 \\ a_2 + b_2 x + c_2 x^2 + d_2 x^3 & \text{for } t_1 \leq x < t_2 \\ \vdots & \\ a_k + b_k x + c_k x^2 + d_k x^3 & \text{for } x \geq t_{k-1} \end{cases}$$

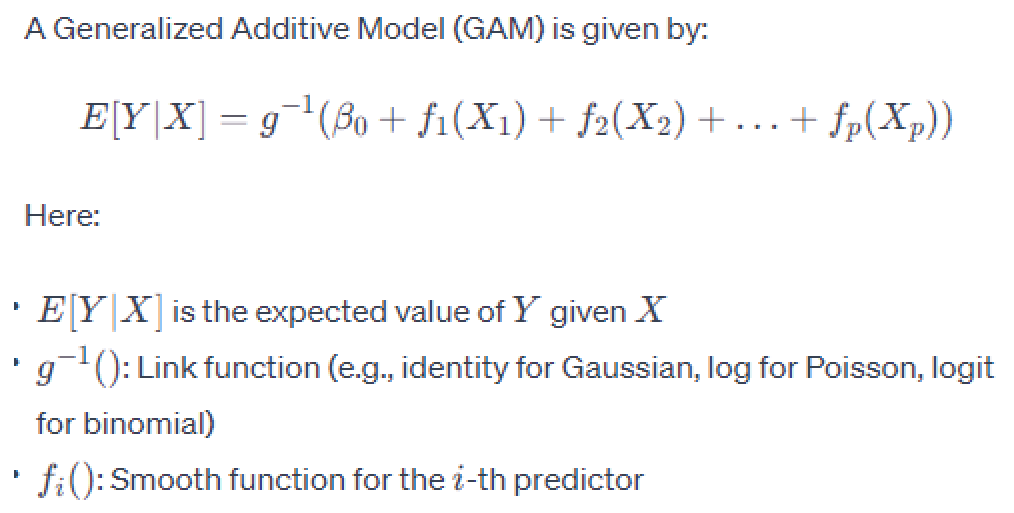
Generalized additive models (GAMs) provide a general framework for extending a standard linear model by allowing non-linear functions of each of the variables, while maintaining additivity.

Just like linear models, GAMs can be applied with both quantitative and qualitative responses.

$$s(x) = \begin{cases} a_1 + b_1 x + c_1 x^2 + d_1 x^3 & \text{for } x < t_1 \\ a_2 + b_2 x + c_2 x^2 + d_2 x^3 & \text{for } t_1 \leq x < t_2 \\ \vdots & \\ a_k + b_k x + c_k x^2 + d_k x^3 & \text{for } x \geq t_{k-1} \end{cases}$$

Imagine a regression equation where instead of a linear function describing each predictor's relationship to the outcome, you have a smooth, flexible curve, perhaps modeled using splines.

The idea is to allow each predictor its own customized smooth function, which can capture the unique, potentially non-linear, relationship it has with the response variable.



A Generalized Additive Model (GAM) is given by:

$$E[Y|X] = g^{-1}(\beta_0 + f_1(X_1) + f_2(X_2) + \ldots + f_p(X_p))$$

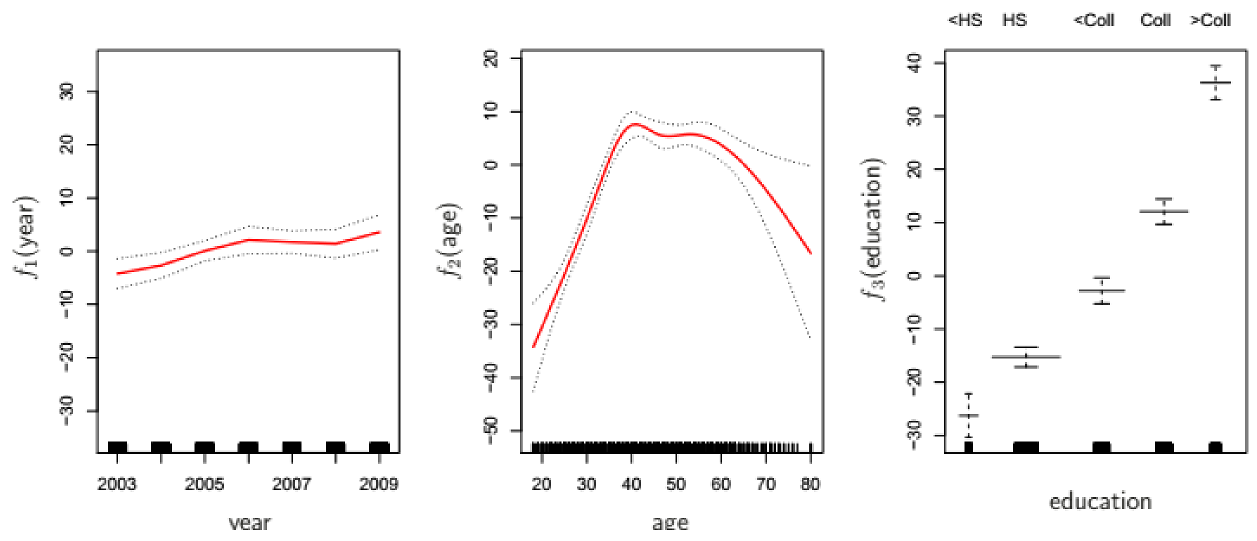
Here:

- $E[Y|X]$ is the expected value of Y given X
- $g^{-1}()$: Link function (e.g., identity for Gaussian, log for Poisson, logit for binomial)
- $f_i()$: Smooth function for the i-th predictor

The function *fi* can be a spline, a polynomial, a local regression, or any other smooth function. Splines are often used due to their flexibility and computational efficiency.

The term "additive" comes from the fact that the model is a sum of smooth functions.

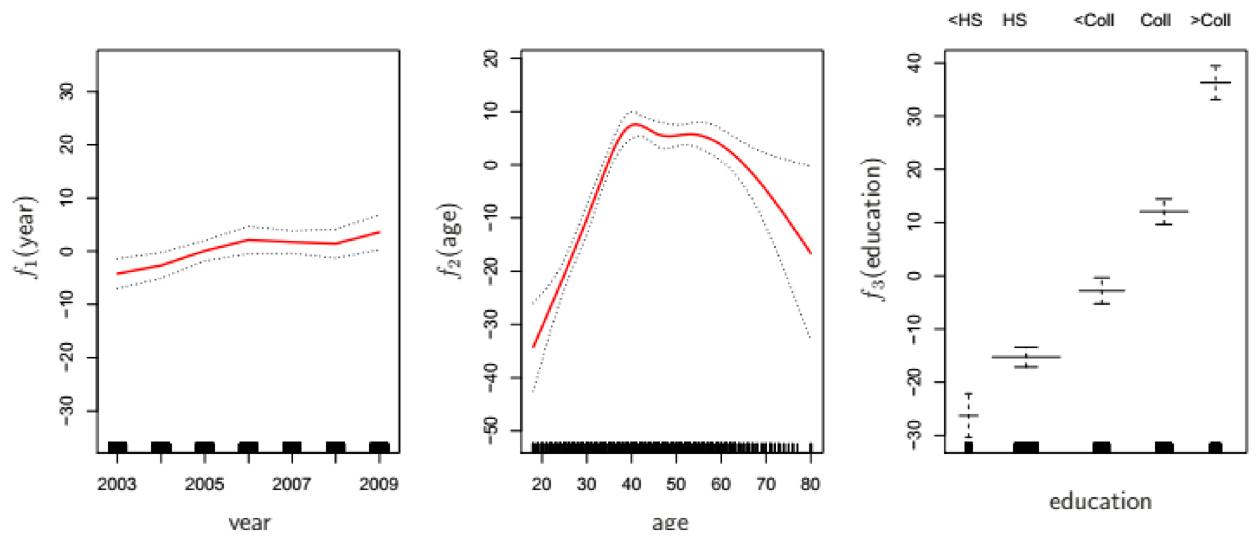
Unlike traditional interaction terms in linear models, **GAMs assume that these smooth functions operate independently, i.e., no interaction terms.**



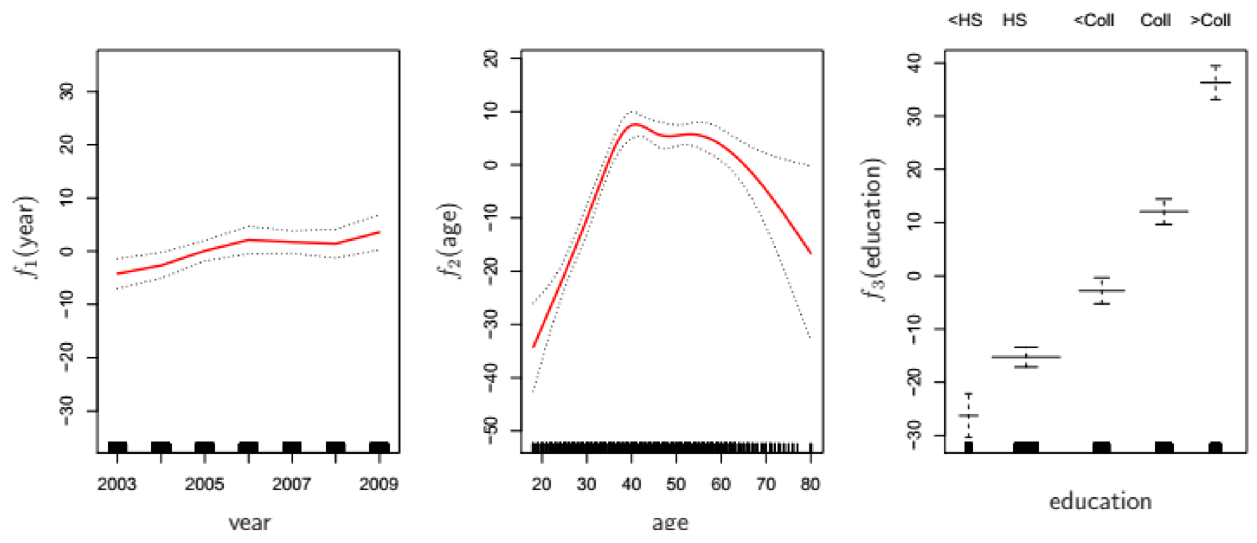
GAMs allow us to fit a non-linear fj to each Xj, so that we can automatically model non-linear relationships that standard linear regression will miss.

This means that we do not need to manually try out many different transformations on each variable individually.

- The non-linear fits can potentially make more accurate predictions for the response Y.
- Because the model is additive, we can examine the effect of each Xj on Y individually while holding all of the other variables fixed.
- The smoothness of the function fj for the variable Xj can be summarized via degrees of freedom.

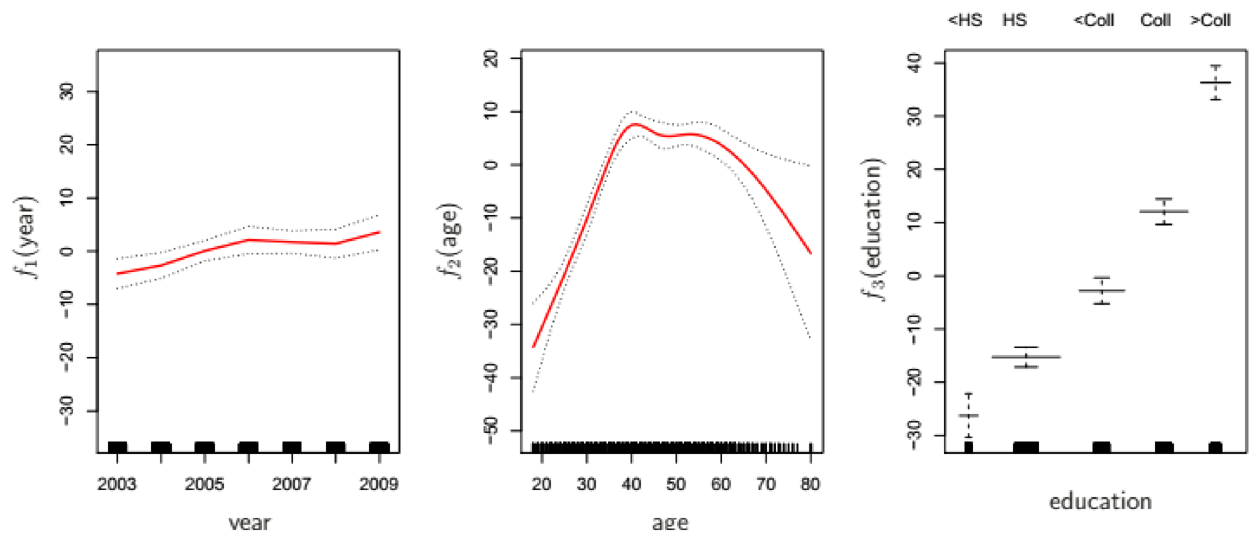


- The main limitation of GAMs is that the model is restricted to be additive. With many variables, **important interactions can be missed**.
- As with linear regression, we can **manually add interaction terms** to the GAM model by including additional predictors of the form Xj × Xk (multiplying features).

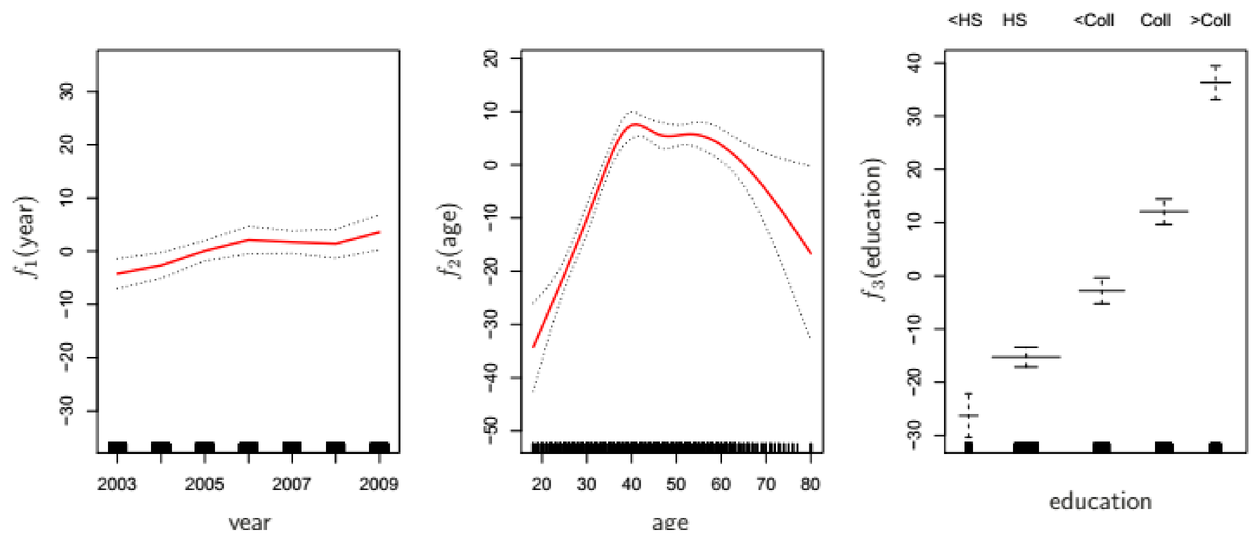


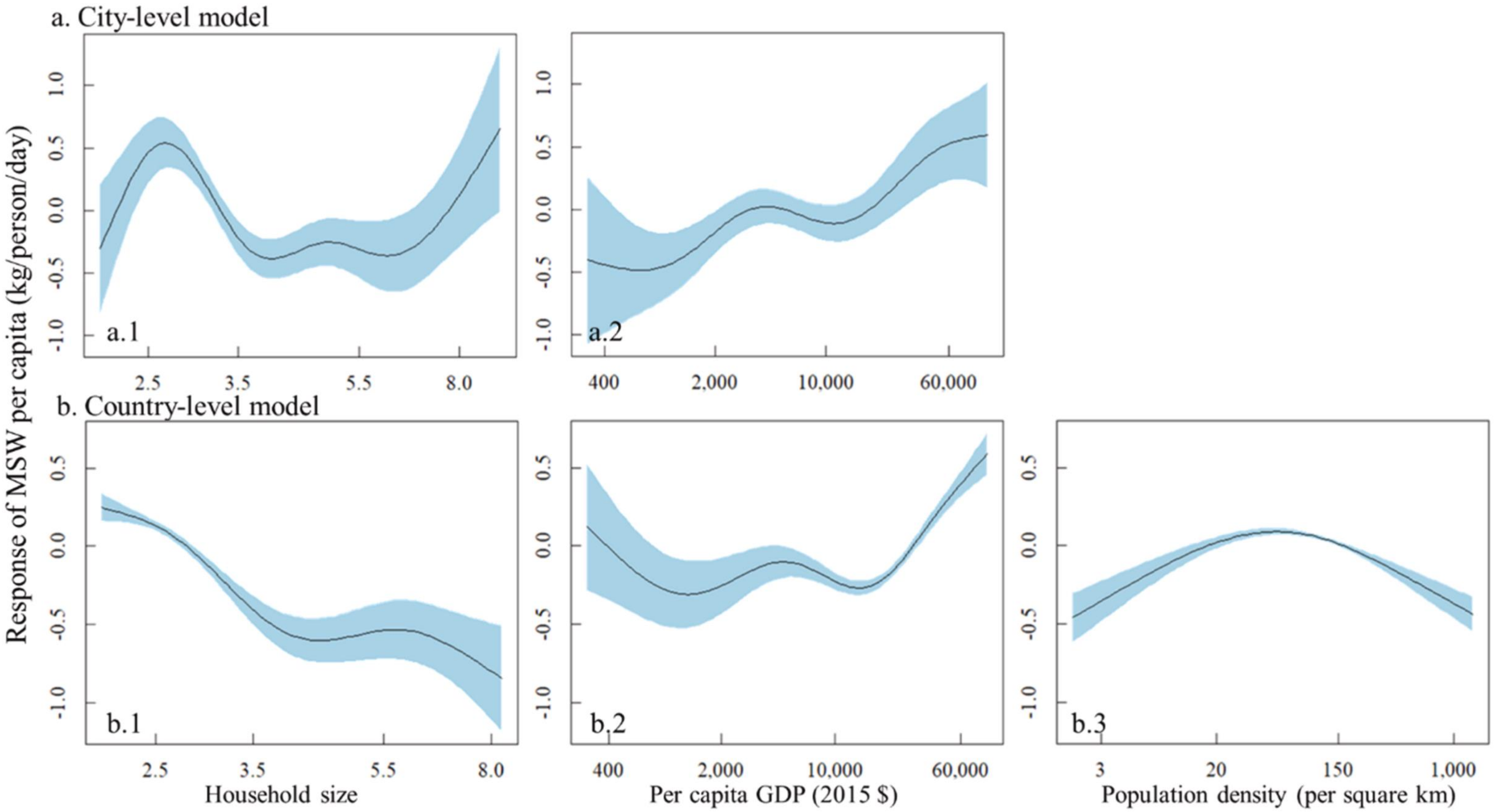
Categorical variables **can** interact with continuous variables.

- For instance, a GAM could include a smooth term that varies by category, modeling a different smooth relationship for each category.
- We need to specifically tell the model to vary one or many relationships by a specific category.

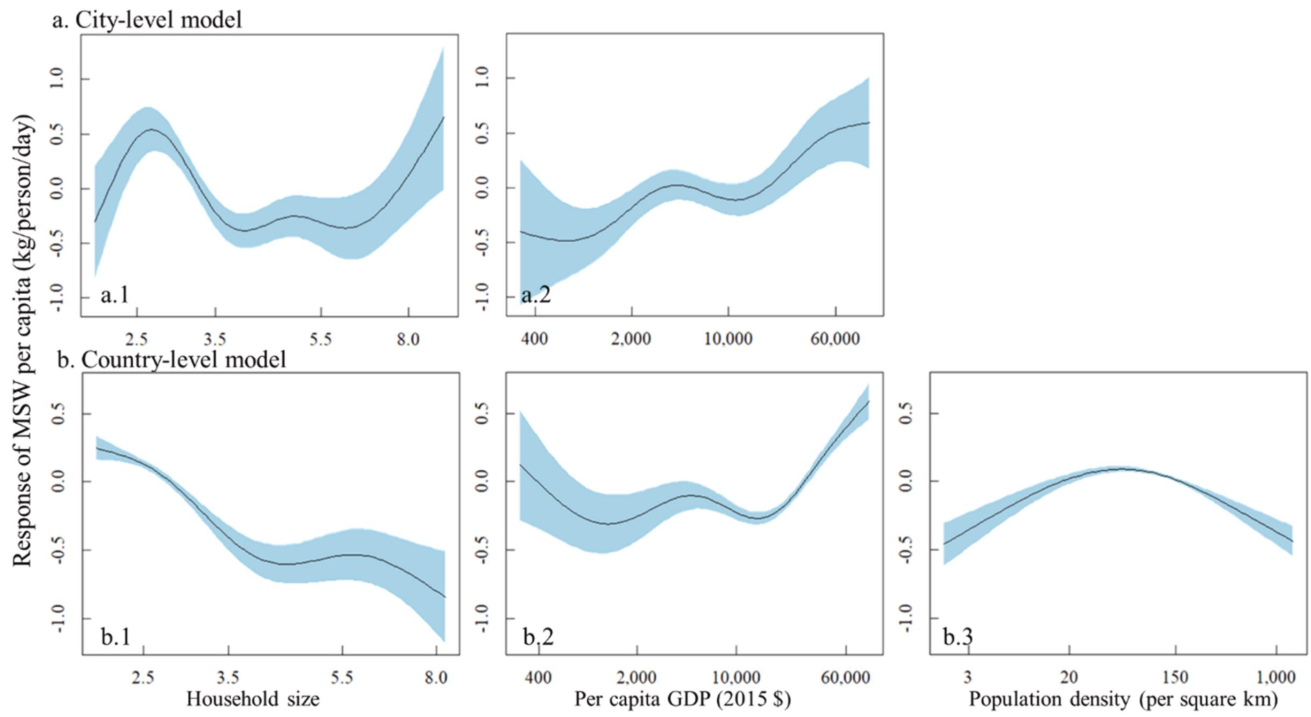


- We can find very interesting **non-linear** relationships for **inference**.
- We have very high risk of **overfitting**.
- Other models are better at automatically finding interactions.




a. City-level model
b. Country-level model
Response of MSW per capita (kg/person/day)
a.1
a.2
b.1
b.2
b.3
1.0
0.5
0.0
-0.5
-1.0
2.5
3.5
5.5
8.0
400
2,000
10,000
60,000
3
20
150
1,000
Household size
Per capita GDP (2015 $)
Population density (per square km)

- lab


a. City-level model
Response of MSW per capita (kg/person/day)
1.0
0.5
0.0
-0.5
-1.0
a.1
2.5
3.5
5.5
8.0
a.2
400
2,000
10,000
60,000
b. Country-level model
0.5
0.0
-0.5
-1.0
b.1
b.2
b.3
Household size
Per capita GDP (2015 $)
3
20
150
1,000
Population density (per square km)