## Regression

Regression is a statistical method that models the relationship between a dependent variable and one or more independent variables

### How does regression ML work?

Goal: develop a model function to predict an output given some input

Steps:

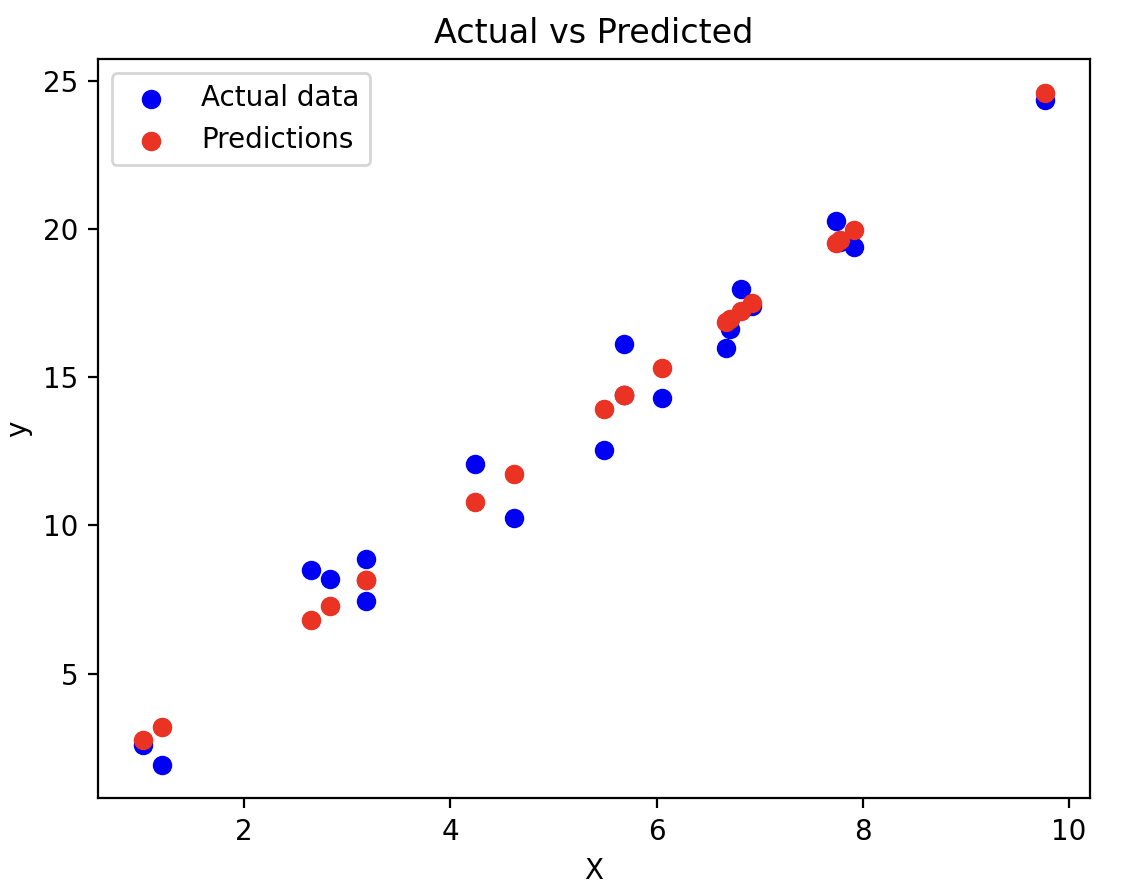
1. **Input**: take input data
2. **Model** function: the model predicts an output value (e.g. using linear equations)
3. **Compare/evaluate**: a cost function (e.g. MSE) quantifies the prediction error
4. **Feedback/optimization**: parameters of the model function are adjusted (e.g. using gradient descent) based on the error to improve future predictions

We loop through these steps to train a model

### Basic Types of Regression

#### Linear Regression

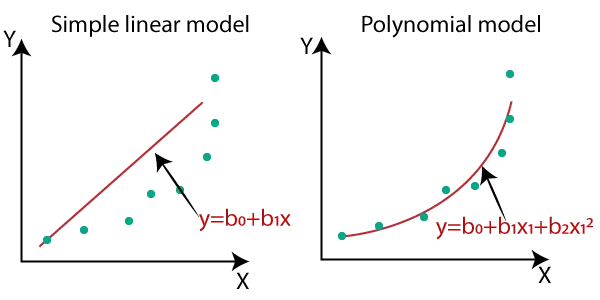
* Y = mx + b + e, where e is the error
  + The model is like a line of best fit
* Pros: easy to implement and train
* Cons: assumes linear relationship, cannot have multicollinearity (when independent variables are dependent on each other)
* [Github example: linear regression](https://github.com/sherryliu-lsy/Regression/blob/main/ex_linear_regression.py)



* **Multi-linear regression** - multiple independent variables and one dependent variable

#### Polynomial Regression

* Similarly, the curve can be modeled as an nth degree polynomial
* Generally used when linear regression is unable to describe the result clearly
* y = a0 + a1x + a2x2 .. + anxn
* Pros: can capture nonlinear relationships



* Cons: increasing degree of the polynomial can lead to overfitting, poor extrapolation

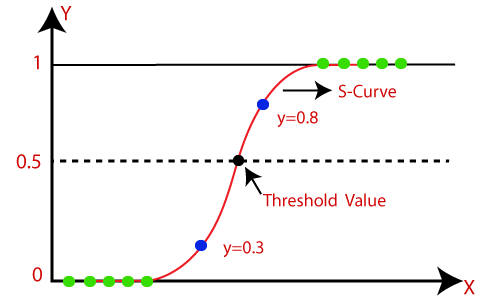
#### Stepwise Regression

* To build an accurate and parsimonious (i.e. explaining data using fewest variables) model

1. **Forward Selection** – start with an empty model and iteratively add variables
2. **Backward Elimination** – start with a model that includes all variables and iteratively remove variables

#### Logistic Regression

* Outputs probabilities using the **sigmoid function**, which maps a real-valued set of independent variables input into a value between 0 and 1



* A threshold is set (usually 0.5) to assign input to one class or another
* Often used for binary classification tasks
* Pros: effective for binary classification, simple and interpretable
* Cons: assumes linear relationship, assumes that the features are independent from each other, not as efficient for multi-class problems
* Types of logistic regression:
  + **Binomial** (2 types)
  + **Multinomial** (3+ unordered types, e.g. dolphin/dog/wombat)
  + **Ordinal** (3+ ordered types, e.g. low/medium/high)

### **Advanced Types of Regression**

Used for more complex situations and use more complex optimization techniques (improved accuracy)

#### Ridge and Lasso Regression

* Techniques used to improve prediction accuracy ; variations on linear regression
* **Lasso regression**: form of regularization that tries to minimize the magnitude of coefficients to ensure that more relevant variables are included
  + **Regularization** - reduces a model’s variance by penalizing training input parameters contributing to noise
* **Ridge regression**: introduces additional term that penalizes large coefficient values

#### Gaussian Process Regression (GPR)

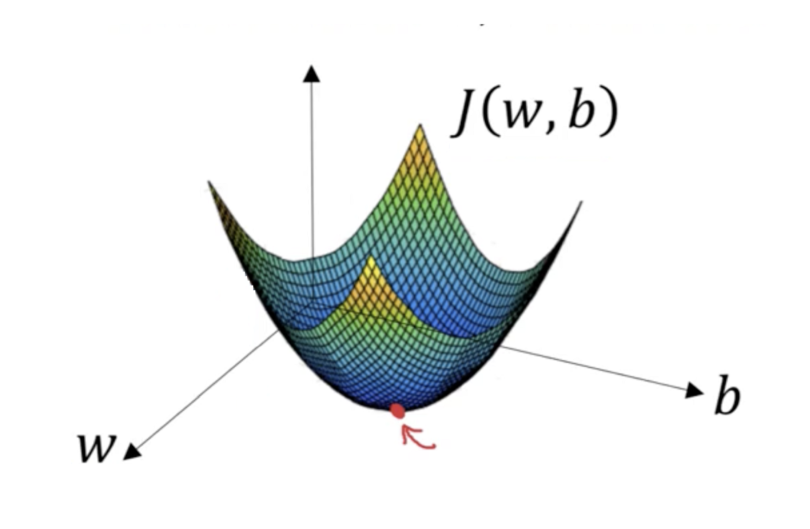
* Bayesian method capable of performing both classification and regression
* Often used for predicting continuous output values
* non-parametric regression technique -> means that there is no fixed form or structure for the relationship between input variables and output variable (target); no set number of model parameters
* Steps for the Process
  + Gather input-output data pairs
  + Select a **Kernel/Covariance Function**
    - Covariance function: outputs a scalar value that represents the similarity between two input points (higher = more similar)
  + Parameter Optimization: estimate the **hyperparameters** of the kernel function by maximizing the likelihood of the data
    - Hyperparameters = configuration setting defined before training for ex. degree of polynomial in polynomial regression or learning rate
* Advantages:
  + useful for irregularly sampled data because they are good at smoothing noisy data
  + adjusts to complexity of the data since there is no set number of model parameters
  + can also provide uncertainty estimates, especially for smaller datasets
* Cons:
  + GPR performance is very dependent on the choice of Kernel function, so our choice may lead to poor predictions or overfitting
  + requires careful tuning of kernel hyperparameters
  + assumes that the noise in the data is Gaussian (normally distributed)
* Applications
  + Anomaly Detection: this is a good application since anomalies are often points of high uncertainty; sensitive to outliers; can detect points that break covariance structure
  + Some real life examples include anomaly detection for insertion tasks in robotic assembly, sparse gamma-ray data, etc.

More advanced types of regression:

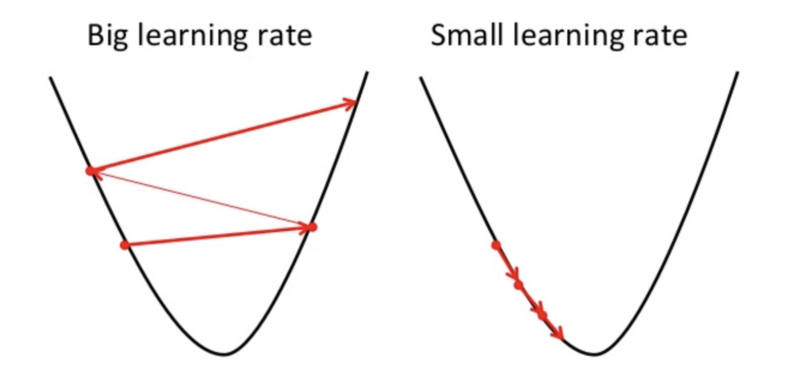
* Support Vector Regression (SVR)
* Generalized Additive Models (GAM)
* Elastic Net
  + combination of Ridge and Lasso
* Bayesian Regression

### Gradient Descent (Optimization Algorithm)

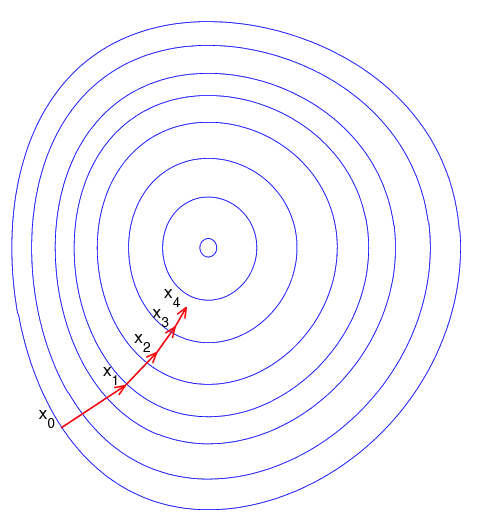
* Optimization algorithm for finding a **local minimum of a differentiable function**
  + Set initial parameters
  + Use calculus to iteratively adjust the parameter values so they minimize the given cost-function
  + Picture parameters being the dimensions, we want to find the minimum of the cost function



* **Gradient** - measures the change in all weights with respect to the change in error
  + Like the slope of a function. The more the gradient, the faster the model learns
  + In math language, a gradient is a partial derivative with respect to inputs
  + Gradient shouldn’t be too large or too small



* Analogy

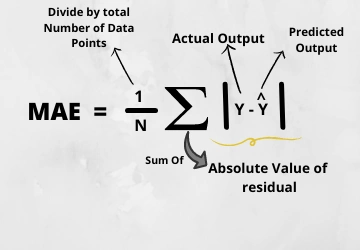


* + We want to climb up a hill (blindfolded) and reach the top with as few steps as possible. We can take bigger steps in the steepest direction, but as we get closer to the top, take smaller steps to avoid overshooting
  + A gradient here is a vector that contains the direction and length of the steepest step we can take

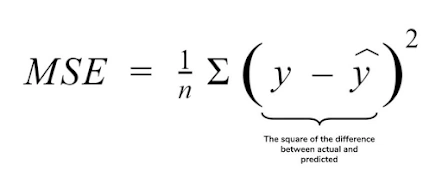
### Cost Functions

Cost functions evaluate the model **during training**.

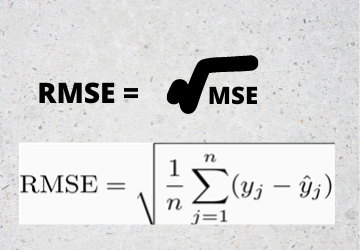
* **Mean Absolute Error (MAE)**



* + The MAE you get is in the same unit as the output variable.
  + Most robust to outliers.
* **Mean Square Error (MSE)**



* + Benefits of squaring the error (why MSE better than MAE):
    - Avoids the cancellation of negative terms
    - Punishes larger errors, but disadvantage: not robust to outliers
  + Result is differentiable, unlike MAE
    - We need the result of the cost function to be differentiable so we can use calculus (e.g. gradient descent) to find the minimum cost
* **Root Mean Square Error (RMSE)**



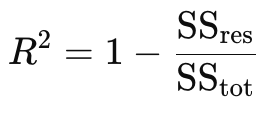
* + Even more popular than MSE because the result is in the y-value units
  + But also not robust to outliers
* **Root Mean Square Log Error (RMSLE)**
  + Taking the log of RMSE slows down the scale of error and mitigates impact of outliers
* [Example: MAE, MSE, RMSE, RMSLE](https://github.com/sherryliu-lsy/Regression/blob/main/ex_cost_functions.py)

### Model Evaluation

#### Metrics

These metrics evaluate the model **after it’s trained**. They are independent of context, unlike cost functions.

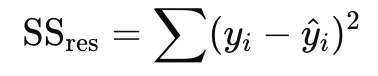
* **R Squared (R2)**



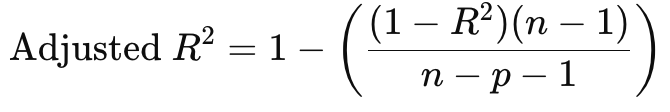
* + Aka “coefficient of determination”, “goodness of fit”
  + Shows how well your regression model explains the variability of the target variable, from 0 to 1 (e.g. 1 = predictions fit tests perfectly, 0 = doesn’t fit at all)
  + **Total sum of squares** - measures the total deviation of the actual values from the mean of the actual values



* + **Residual sum of squares** - measures the total deviation of the predicted values from the actual values



* **Adjusted R Squared (Adjusted R2)**



* + n = number of test/data points/observations
  + p = number of predictors (independent variables)
  + Improves R2: accounts for the number of independent variables (predictors)
    - R2 can increase when you add more predictors, even if they don’t actually improve the model
    - Adjusted R2 penalizes the R2 value based on the number of predictors, so only meaningful predictors contribute to the score

#### Overfitting vs Underfitting

* High-bias models oversimplify data.
* High variance models over-adapt to data.
* **Overfitting**: high variance, low bias
  + To prevent overfitting:
    - Increase volume of training data
    - Data augmentation (modify training data strategically)
    - Halt training when necessary (avoid model focusing on small details)
* **Underfitting**: low variance, high bias
  + To prevent underfitting:
    - Increase training time
    - Optimize model complexity
    - Lower regularization

## Interesting Problems to Solve with Regression

* likelihood of a person developing a certain disease based on their genetic makeup
* Image classification (aka classifying the objects in an image)
* Estimating the age of a person in an image apparently
* Sleeping time
* Salary vs experience (work/academic), industry/field

| Inputs | Output | Type of Regression |
| --- | --- | --- |
| Genetic makeup | Likelihood of developing a certain disease | multinomial Logistic Regression? |
| * Day in the semester * Course load * Program * Term (e.g. 2A) | Sleeping schedule of UW students | multivariate linear, polynomial |
| Experience (work/academic/extracurricular), industry/field | Salary | Linear regression? |
| Image | Classified object | Logistic regression |

## Sources

* <https://datacouch.medium.com/solving-real-world-problems-using-regression-models-9fae03c940af>
* [Overfitting vs Underfitting](https://www.coursera.org/articles/overfitting-vs-underfitting?utm_medium=sem&utm_source=gg&utm_campaign=B2C_NAMER__coursera_FTCOF_career-academy_pmax-enhanced-NRL-w/in-14d-new-cust-country-US-country-CA&campaignid=20397118025&adgroupid=6472952357&device=c&keyword=&matchtype=&network=x&devicemodel=&adposition=&creativeid=6472952357&hide_mobile_promo&gad_source=1&gclid=Cj0KCQjwjNS3BhChARIsAOxBM6oxXQU5Fa5OQuhufcOZgN-wXTQRgZfR4zQCtno_Y4ct-slJEss_ZUoaAkq-EALw_wcB)
* <https://www.geeksforgeeks.org/understanding-logistic-regression/>
* <https://medium.com/@byanalytixlabs/what-are-lasso-and-ridge-techniques-05c7f6630f6b>
* <https://www.geeksforgeeks.org/gaussian-process-regression-gpr/>
* <https://medium.com/data-science-at-microsoft/introduction-to-gaussian-process-regression-part-1-the-basics-3cb79d9f155f>
* GPR Application Example: <https://merl.com/publications/docs/TR2019-055.pdf>
* Model evaluation: <https://www.analyticsvidhya.com/blog/2021/05/know-the-best-evaluation-metrics-for-your-regression-model/#:~:text=Common%20regression%20evaluation%20metrics%20for,Absolute%20Percentage%20Error%20(MAPE)>.
* Gradient descent: <https://builtin.com/data-science/gradient-descent#:~:text=What%20Is%20Gradient%20Descent%20in,function%20as%20far%20as%20possible>.
* And last but not least, our very chatty friend: <https://chatgpt.com/>

## Tasks

1. Read about regression technique
2. Find an interesting problem to solve with regression
3. Find out how you measure, quality of your model? What’s the quality of the model means? define metrics? and etc.
4. More advanced regression techniques, maybe one or two enough
5. Show challenges of technique (pros and cons) and its variations and some advanced example for each.