

Machine Learning

Interview Preparation



1. What is Data Science ?
2. What does Data Scientist do ?



DATA SCIENCE

- Data Science is a blend of various tools, algorithms, and machine learning principles with the goal to discover hidden patterns from the raw data. How is this different from what statisticians have been doing for years? The answer lies in the difference between explaining and predicting: statisticians work a posteriori, explaining the results and designing a plan; data scientists use historical data to make predictions.



3. What is Re-Sampling?
4. How do you use re-sampling in ML?



Re-sampling:

- Resampling is a methodology of economically using a data sample to improve the accuracy and quantify the uncertainty of a population parameter. Resampling methods, in fact, make use of a nested resampling method.
- Two commonly used resampling methods that you may encounter are **k-fold cross-validation** and the **bootstrap**.



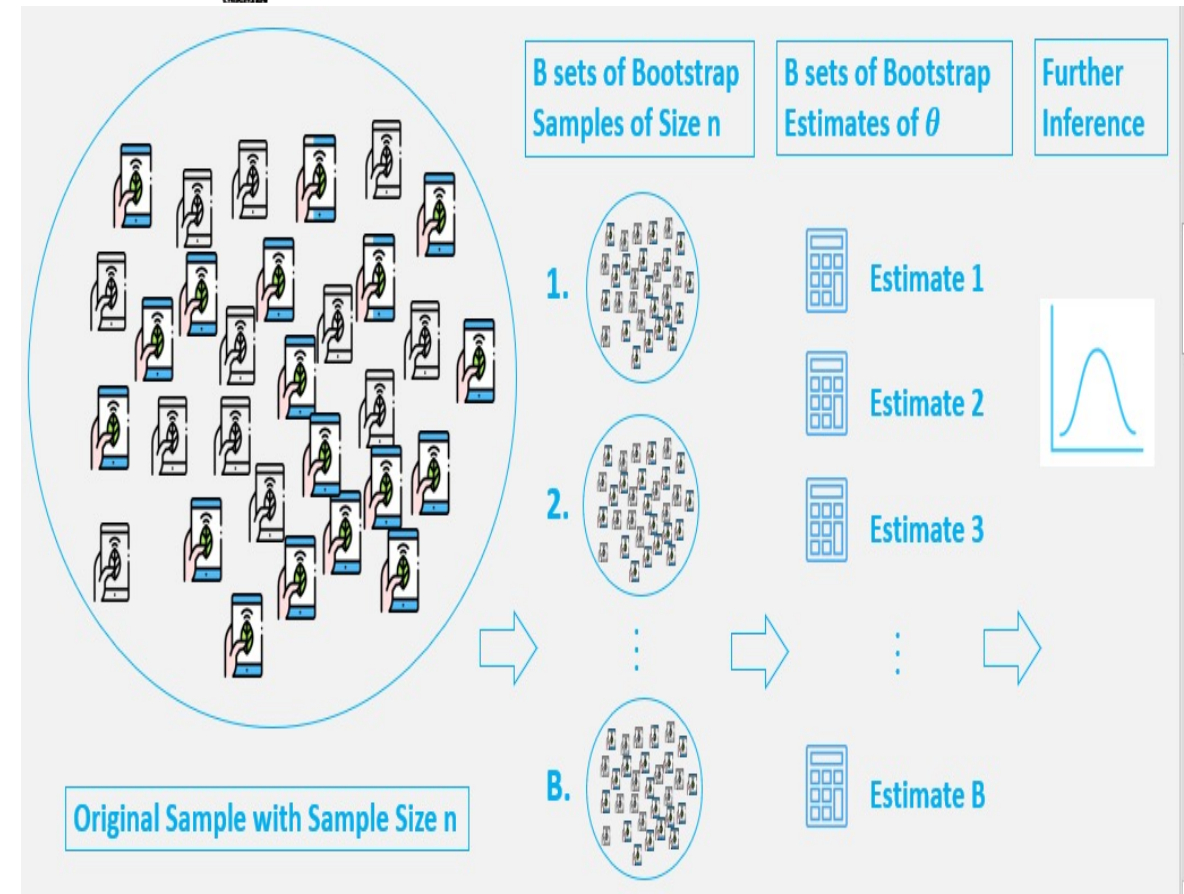
5. Would you explain in your words, what is **Bootstrap**?

6. How do you use **Bootstrap** in ML?



Bootstrap

- Samples are drawn from the dataset with replacement (allowing the same sample to appear more than once in the sample), where those instances not drawn into the data sample may be used for the test set.



7. Would you explain in your words, what is **K-Fold Cross Validation**?

8. How do you use **K-Fold Cross Validation** in ML?

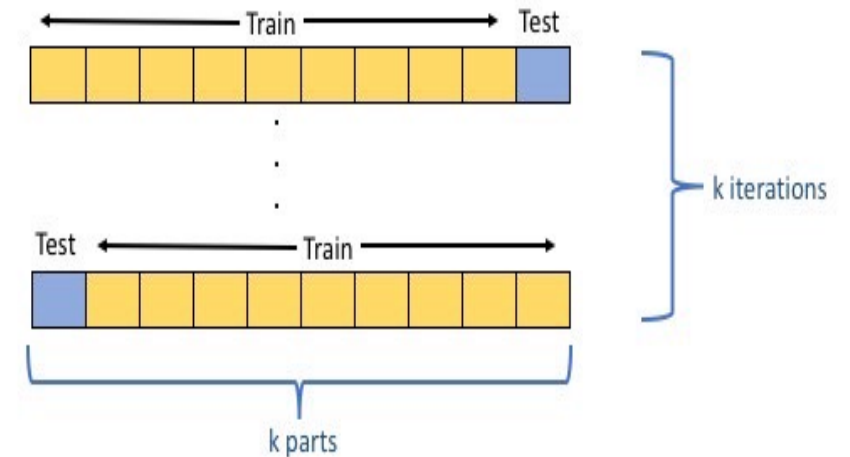


K-fold Cross-validation

- A dataset is partitioned into k groups, where each group is given the opportunity of being used as a held out test set leaving the remaining groups as the training set.
- The k -fold cross-validation method specifically lends itself to use in the evaluation of predictive models that are repeatedly trained on one subset of the data and evaluated on a second held-out subset of the data.
- There is an alternative in Scikit-Learn called **Stratified k fold**, in which the split is shuffled to make it sure you have a representative sample of each class and a k fold in which you may not have the assurance of it (not good with a very unbalanced dataset).

K Folds Cross Validation Method

1. Divide the sample data into k parts.
2. Use $k-1$ of the parts for training, and 1 for testing.
3. Repeat the procedure k times, rotating the test set.
4. Determine an expected performance metric (mean square error, misclassification error rate, confidence interval, or other appropriate metric) based on the results across the iterations



9. Would you explain in your words what is ML?



MACHINE LEARNING

- The primary aim is to allow the computers to learn automatically without human intervention or assistance and adjust actions accordingly. Machine Learning explores the study and construction of algorithms that can learn from and make predictions on data.



9. What is **bias-variance trade-off**?



- **Bias-Variance trade-off**: The goal of any supervised machine learning algorithm is to have low bias and low variance to achieve good prediction performance.
- There is no escaping the relationship between bias and variance in machine learning. Increasing the bias will decrease the variance. Increasing the variance will decrease bias.



Bias

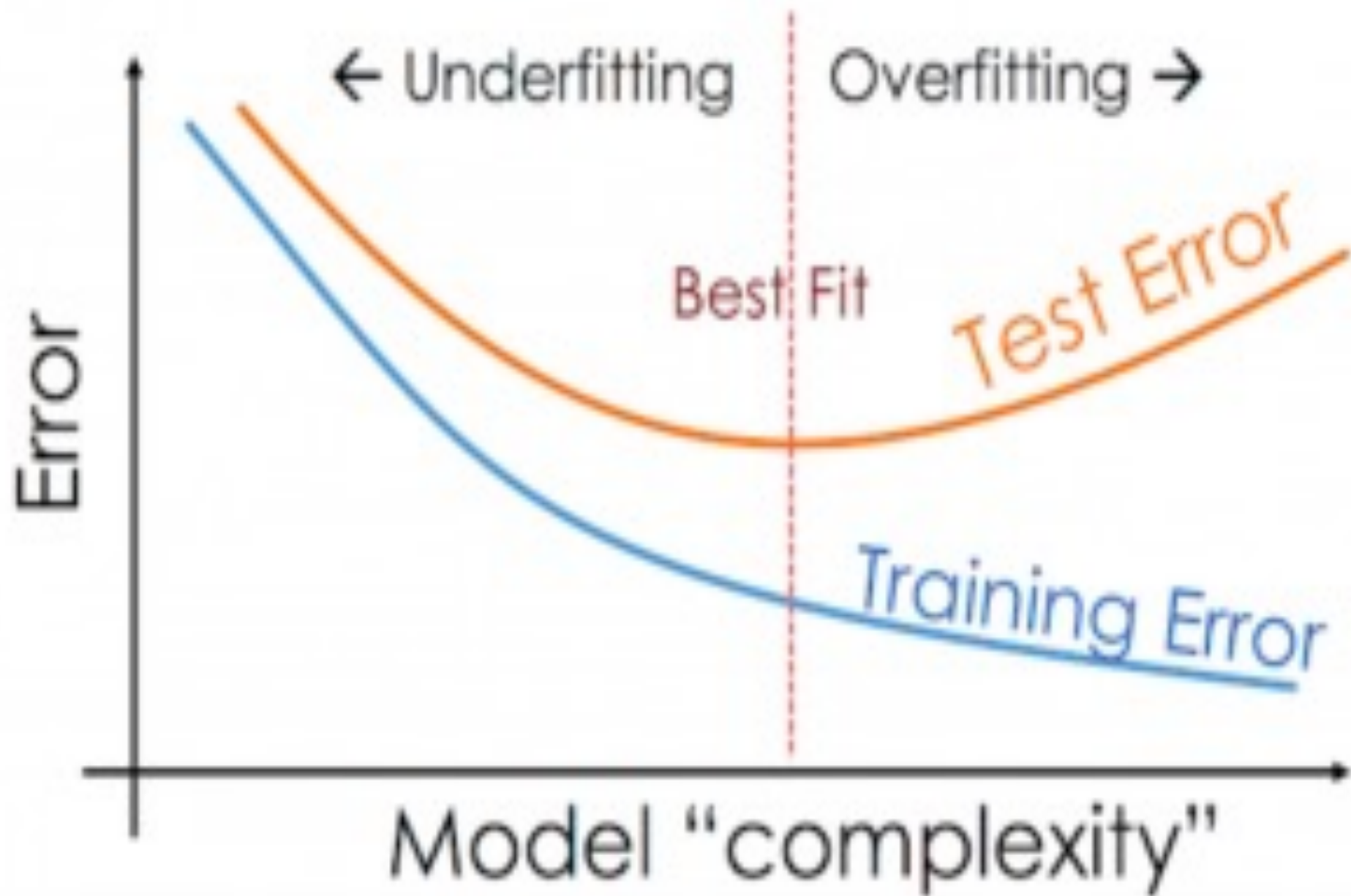
- Bias: Bias is an error introduced in the model due to the oversimplification of the algorithm used (does not fit the data properly). It can lead to under-fitting.
- Low bias machine learning algorithms – Decision Trees, k-NN and SVM
- High bias machine learning algorithms – Linear Regression, Logistic Regression



Variance

- Variance: Variance is error introduced in the model due to a too complex algorithm, it performs very well in the training set but poorly in the test set. It can lead to high sensitivity and overfitting.
- High Variance machine learning algorithms — Decision Trees, k-NN and SVM
- Low Variance machine learning algorithms — Linear Regression, Logistic Regression





10. How to deal with bias-variance with ML models,
please give specific examples?



- 1. The k-nearest neighbor algorithm has low bias and high variance, but the trade-off can be changed by increasing the value of k which increases the number of neighbors that contribute to the prediction and in turn increases the bias of the model.
- 2. The support vector machine algorithm has low bias and high variance, but the trade-off can be changed by increasing the C parameter that influences the number of violations of the margin allowed in the training data which increases the bias but decreases the variance.
- 3. The decision tree has low bias and high variance, you can decrease the depth of the tree or use fewer attributes.
- 4. The linear regression has low variance and high bias, you can increase the number of features or use another regression that better fits the data.



11. What is overfitting?

12. How do you deal with the overfitting in your model?



OVERFITTING

- Overfitting refers to a model that is only set for a very small amount of data and ignores the bigger picture. There are main methods to avoid overfitting:
- Keep the model simple—take fewer variables into account, thereby removing some of the noise in the training data.
- Pruning- Pruning as the name suggests, is cutting. We prune wanted data in addition to nodes that gets easily affected by wrong data.
- Use cross-validation techniques, such as k folds cross-validation
- Use regularization techniques, such as LASSO, that penalize certain model parameters if they're likely to cause overfitting



To Combat Overfitting

- 1. Add noise
- 2. Feature selection
- 3. Increase training set
- 4. L2 (ridge) or L1 (lasso) regularization; L1 drops weights, L2 no
- 5. Use cross-validation techniques, such as k folds cross-validation
- 6. Boosting and bagging
- 7. Dropout technique
- 8. Perform early stopping
- 9. Remove inner layers



13. What is underfitting?

14. How do you deal with the underfitting in your model?



To Combat Underfitting

- 1. Add features
- 2. Increase time of training

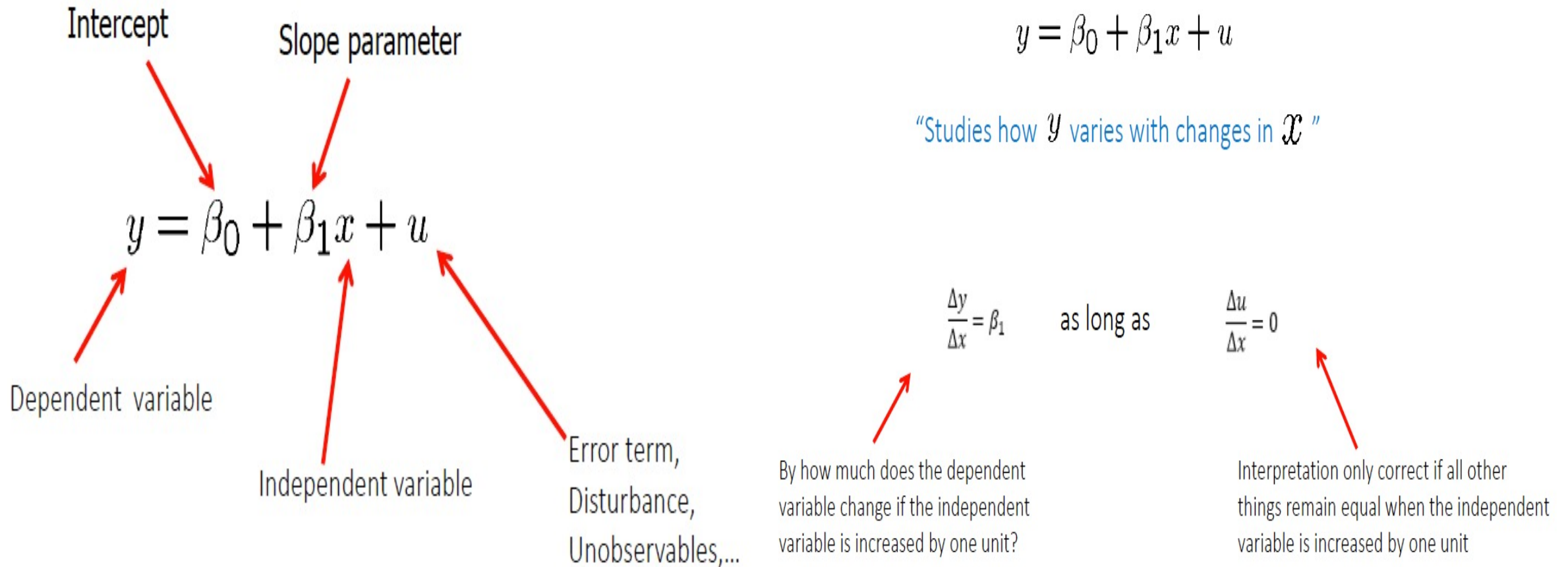


15. What is Regularization?

16. When do you use regularization in your ML models?



The Simple Regression Model



Linear Models

The linear model is an important example of a **parametric** model

- We have a collection of labeled examples $\{(X_i, y_i)\}_{i=1}^N$, where
 - N is the size of the collection
 - X_i is the **D-dimensional** feature vector
 - y_i is a real-valued target

$$f_{w,b}(X) = \mathbf{W}X + \mathbf{b}$$

The model $f_{w,b}$ is a linear combination of features and parameterized by **W** and **b**

- **W** is a D-dimensional vector of parameters
- **b** is a real number

Without regularization: *minimize (Loss (model))*

With regularization: *minimize (Loss (model) + Complexity (model))*



Regularization

- One of the goals of model training is to identify the signal and ignore the noise if the model is given free rein to minimize error, there is a possibility of suffering from overfitting. Regularization imposes some control on this by providing simpler fitting functions over complex ones.
- Regularization is the process of adding tuning parameter (penalty term) to a model to induce smoothness in order to prevent overfitting. This is most often done by adding a constant multiple to an existing weight vector. This constant is often the L1 (Lasso - $|\alpha|$) or L2 (Ridge - $|\alpha|^2$). The model predictions should then minimize the loss function calculated on the regularized training set.
- Regularization becomes necessary when the model begins to overfit/underfit. This technique introduces a cost term for bringing in more features with the objective function. Hence, it tries to push the coefficients for many variables to zero and hence reduce the cost term. This helps to reduce model complexity so that the model can become better at predicting (generalizing).

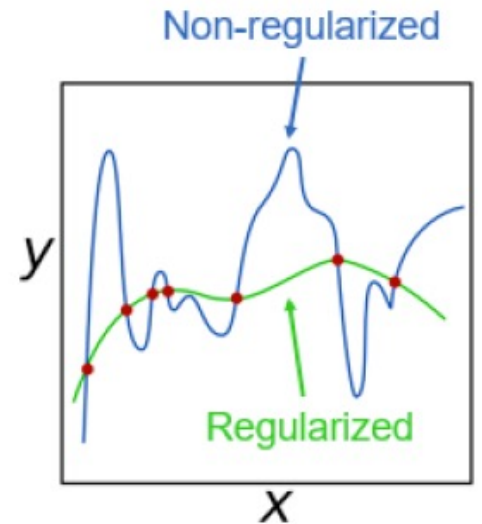


Regularization

- ❑ In machine learning there are often **many features** (usually **correlated** with each other). This can lead to **overfitting** and models that are **unnecessarily complex**.
- ❑ **Regularization** force the learning algorithm to build a **less complex model**. In practice, that often leads to **slightly** higher bias but **significantly** reduces the variance.

✓ The two most widely used types of regularization are called **L1** and **L2** regularization. The idea is quite simple. To create a regularized model, we modify the loss function by adding a penalizing term whose value is higher when the model is more complex.

$$\text{Min}_{w,b} (\text{MSE} + \text{penalty}) = \text{Min} \left[\frac{1}{N} \sum_{i=1}^N \left(y_i - f_{w,b}(X_i) \right)^2 + \text{penalty}(w) \right]$$

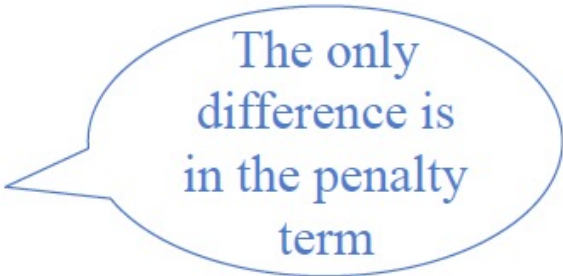


Penalized regression

$$\text{Min}_{w,b} (\text{MSE} + \text{penalty}) = \text{Min} \left[\frac{1}{N} \sum_{i=1}^N \left(y_i - f_{w,b}(X_i) \right)^2 + \text{penalty}(w) \right]$$

- **Penalized regression** is useful for reducing a large number of features to a manageable set and for making good predictions especially where features are correlated (i.e., when classical linear regression breaks down).
- **Penalized regression** can be used to avoid **overfitting**.
- To use the penalized regression, we need to first **standardize the features**. This will allow us to compare the magnitudes of regression coefficients for the feature variables.

- 1) Ridge regression
- 2) LASSO regression
- 3) Elastic Net regression



The only
difference is
in the penalty
term

Norms

- In mathematics, the **norm** of a vector is its **length**.
- In regression analysis, to fit our linear model, we need a measure of **mismatch**!
- Our vector is error at each training data. **We want to measure the length of error!**

- **L1** norm: Least absolute errors

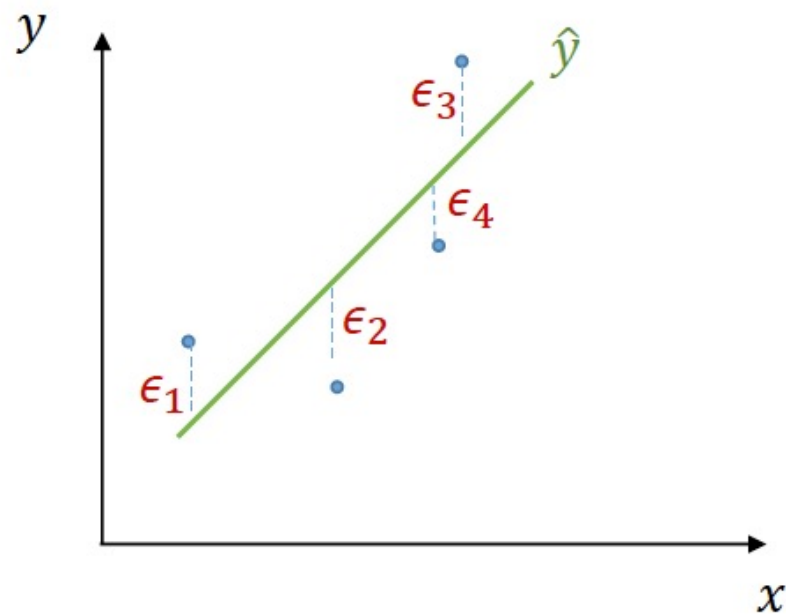
Manhattan norm

$$L^1 = \sum_i |\epsilon_i|$$

- **L2** norm: Least squares

Euclidean norm

$$L^2 = \sum_i (\epsilon_i)^2$$



Similarity/Dissimilarity Metrics

Distance functions

Euclidean

$$\sqrt{\sum_{i=1}^k (x_i - y_i)^2}$$

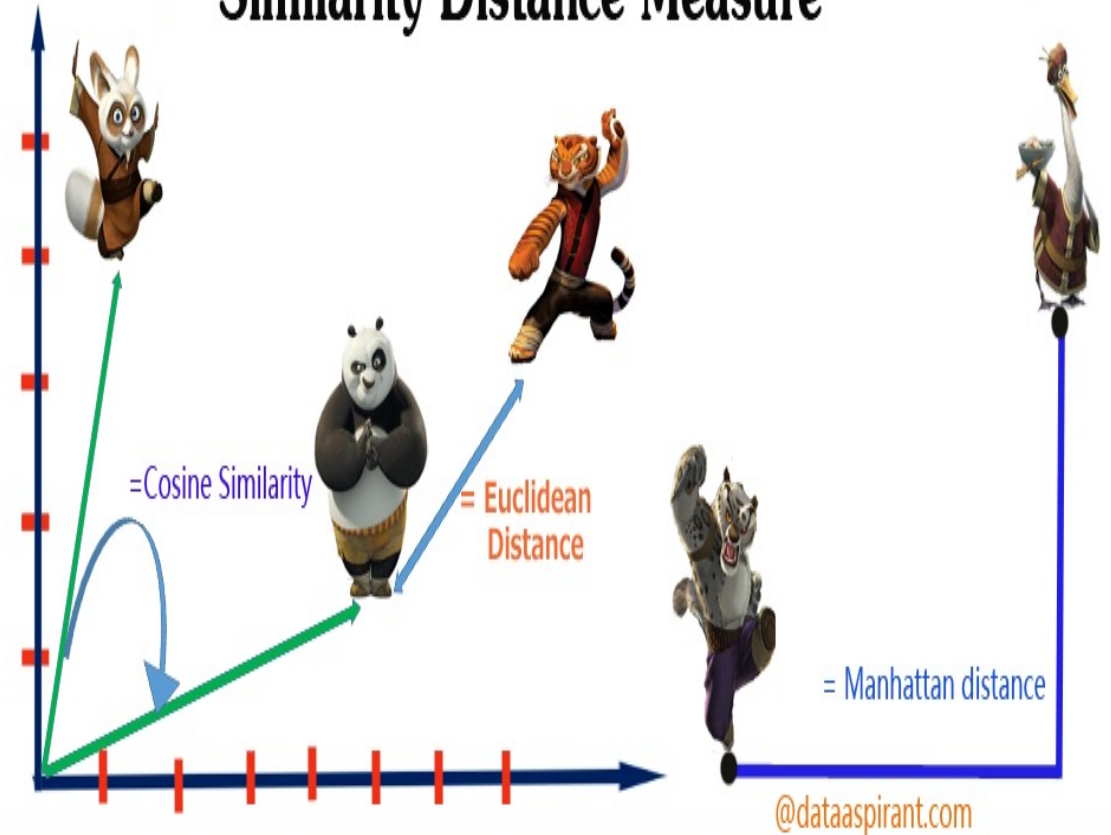
Manhattan

$$\sum_{i=1}^k |x_i - y_i|$$

Minkowski

$$\left(\sum_{i=1}^k (|x_i - y_i|)^q \right)^{1/q}$$

Similarity Distance Measure



17. What is *L1 Regularization*?

18. When do you prefer to use L1 regularization?



L1 Regularization (L1 = lasso)

- Lasso Regression (Least Absolute Shrinkage and Selection Operator) adds “**Absolute value of magnitude**” of coefficient, as penalty term to the loss function.
- Lasso shrinks the less important feature's coefficient to zero; thus, removing some feature altogether. So, this works well for feature selection in case we have a huge number of features.

Cost Function = Loss + L1 Weight Penalty

$$= \underbrace{\sum_{i=1}^M \left(y_i - \sum_{j=1}^N x_{ij} w_j \right)^2}_{\text{Squared Error}} + \underbrace{\lambda \sum_{j=1}^N |w_j|}_{\text{L1 Regularization Term}}$$



19. What is L2 Regularization?

20. When do you prefer to use L2 regularization?



L2 Regularization(L2 = Ridge Regression)

- Regularization adds the penalty as model complexity increases. The regularization parameter (lambda) penalizes all the parameters except intercept so that the model generalizes the data and won't overfit.
- Ridge regression adds “**squared magnitude of the coefficient**” as penalty term to the loss function.
- If lambda is zero, then it is equivalent to OLS. But if lambda is very large, then it will add too much weight, and it will lead to under-fitting. Ridge regularization forces the weights to be small but does not make them zero and does not give the sparse solution. Ridge is not robust to outliers as square terms blow up the error differences of the outliers, and the regularization term tries to fix it by penalizing the weights
- Ridge regression performs better when all the input features influence the output, and all with weights are of roughly equal size. L2 regularization can learn complex data patterns.

$$\text{Cost} = \underbrace{\sum_{i=0}^N (y_i - \sum_{j=0}^M x_{ij} W_j)^2}_{\text{Loss function}} + \lambda \underbrace{\sum_{j=0}^M W_j^2}_{\text{Regularization Term}}$$



21. Can you briefly explain what are differences between L1 - L2 Regularizations?



L2 regularization	L1 regularization
Computational efficient due to having analytical solutions	Computational inefficient on non-sparse cases
Non-sparse outputs	Sparse outputs
No feature selection	Built-in feature selection

L1 Regularization

$$\text{Cost} = \sum_{i=0}^N (y_i - \sum_{j=0}^M x_{ij} W_j)^2 + \lambda \sum_{j=0}^M |W_j|$$

L2 Regularization

$$\text{Cost} = \underbrace{\sum_{i=0}^N (y_i - \sum_{j=0}^M x_{ij} W_j)^2}_{\text{Loss function}} + \lambda \underbrace{\sum_{j=0}^M W_j^2}_{\text{Regularization Term}}$$



End of the 1st Session

