Machine Learning Fundamentals

upGrad knowledgehut

Learning Objectives

- Describe the training and testing paradigm
- Discuss model evaluation and model performance metric
- Explain imbalanced data
- Explore overfitting and cross validation
- Describe bias and variance trade-offs
- Discuss hyperparameters and grid search
- Demonstrate how to implement machine learning model with sklearn using cross-validation and Grid Search

Statistics Vs. Machine Learning

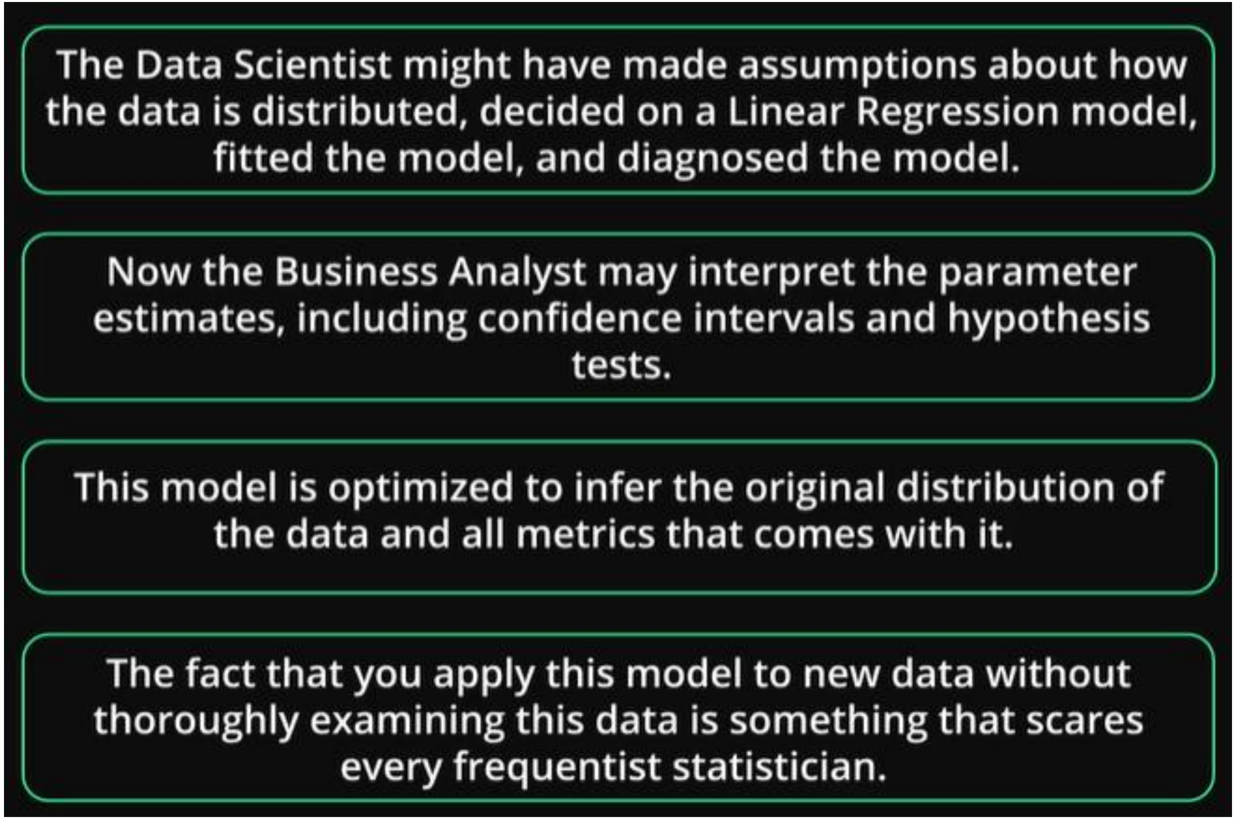
Can the business analyst use the model to accurately predict customer churn?

- Only if the model has a low generalization error and was evaluated properly.

Can the business analyst interpret the p-values to say whether the effect of the demographic information is significant?

- Only if model assumptions hold and relying on a frequentist interpretation of probability.

Statistics Mindset



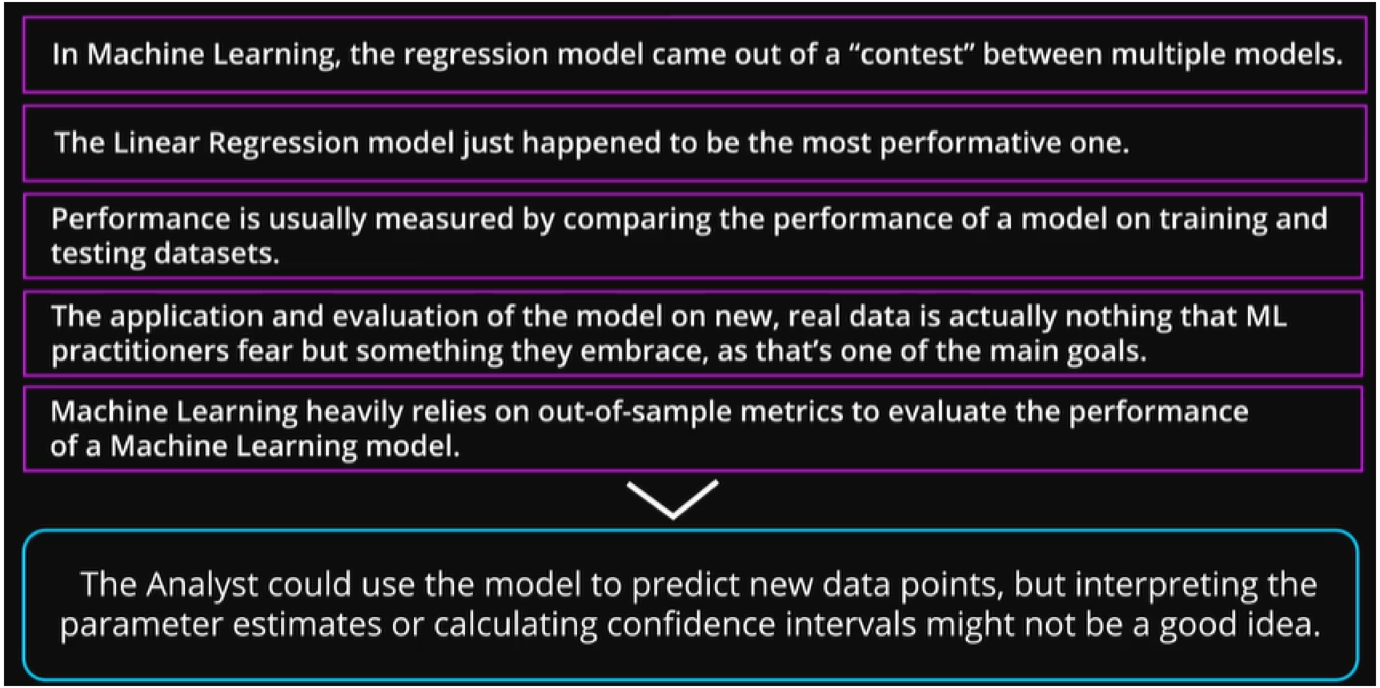
The Data Scientist might have made assumptions about how the data is distributed, decided on a Linear Regression model, fitted the model, and diagnosed the model.

Now the Business Analyst may interpret the parameter estimates, including confidence intervals and hypothesis tests.

This model is optimized to infer the original distribution of the data and all metrics that comes with it.

The fact that you apply this model to new data without thoroughly examining this data is something that scares every frequentist statistician.

Machine Learning Mindsets



In Machine Learning, the regression model came out of a “contest” between multiple models.

The Linear Regression model just happened to be the most performative one.

Performance is usually measured by comparing the performance of a model on training and testing datasets.

The application and evaluation of the model on new, real data is actually nothing that ML practitioners fear but something they embrace, as that’s one of the main goals.

Machine Learning heavily relies on out-of-sample metrics to evaluate the performance of a Machine Learning model.

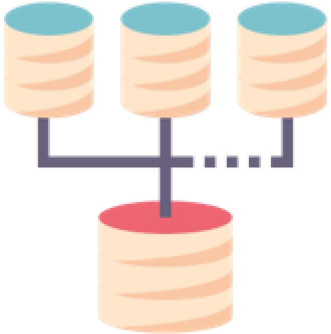
The Analyst could use the model to predict new data points, but interpreting the parameter estimates or calculating confidence intervals might not be a good idea.

Statistics vs. Machine Learning Mindsets

- **Inference**: Statisticians often focus on drawing conclusions about populations based on sample data. They are concerned with understanding the underlying processes that generate the data and making inferences or generalizations from the sample to the larger population.

- **Prediction**: Machine learning practitioners often prioritize building models that can accurately predict outcomes or make decisions based on data. The emphasis is on developing algorithms that can learn patterns from data and generalize to unseen instances.

Machine Learning Mindset



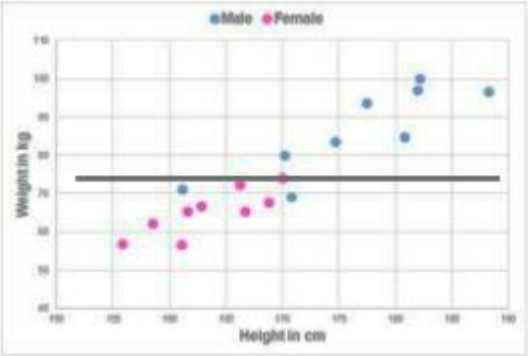
In machine learning, the regression model came out of a "contest" between multiple models.

ML practitioners have nothing to fear from applying and evaluating their models on new, real-world data because it is one of their primary objectives.

Machine learning mainly depends on out-of-sample measurements to assess a machine learning model's performance.

Common Model Classes

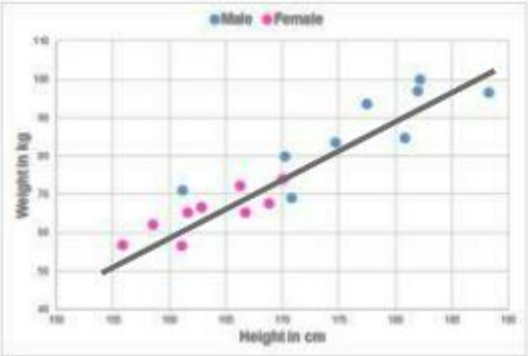
Classification



Which class?

Example: Logistic Regression

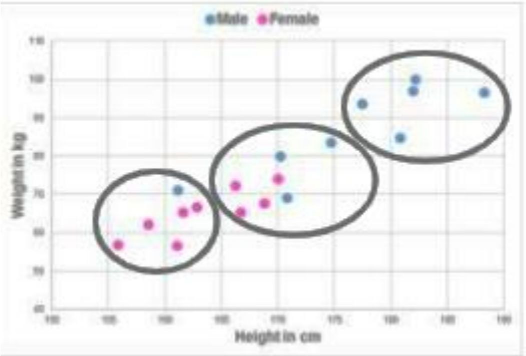
Regression



Which numeric value?

Example: Linear Regression

Clustering

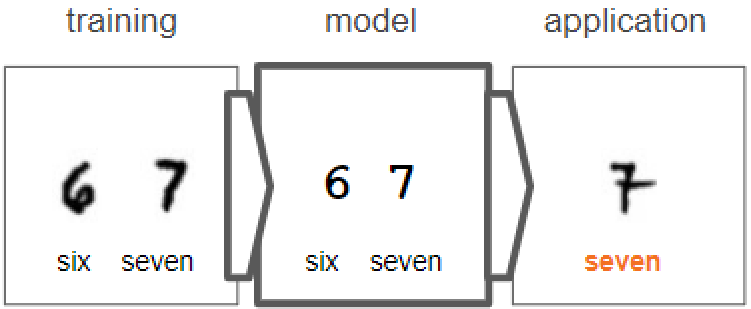


Any groups?

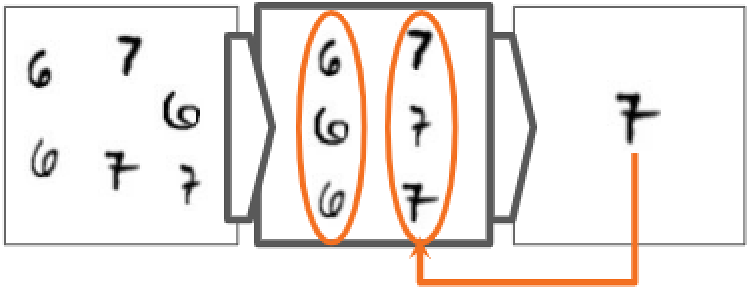
Example: k-Mean Clustering

Supervised vs. Unsupervised Learning

Supervised



Unsupervised



Model Evaluation Methods

Statical Evaluation

Blind test on new data

1. Create model on one part of the data (training set)
2. Validate on an unseen, independent part of the data (validation set)

Calculate a performance metric

1. Accuracy (Recall, Precision, F1-Score)
2. RMSE, R-squared

Improve the performance

1. Adjust data (add/remove features, get more data)
2. Adjust model (change model type and/or hyper parameters)

Statical Evaluation

Training Data

- Used to fit models (estimate parameters).
- Typically, 60-80% of the data set.

Validation Data

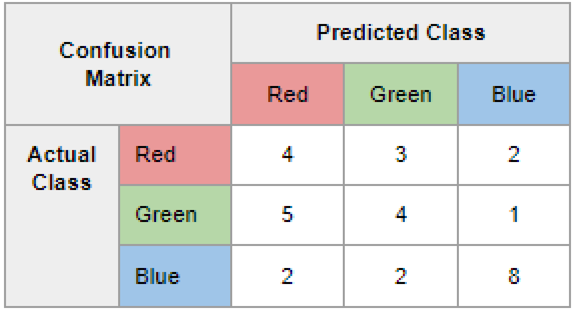
- Used to measure the **error of candidate models**.
- Training and validation are performed iteratively until model achieves the desired performance.

Test Data

- Used to measure the **performance of the final selected model**.
- True blind test as validation data was used multiple times during training.

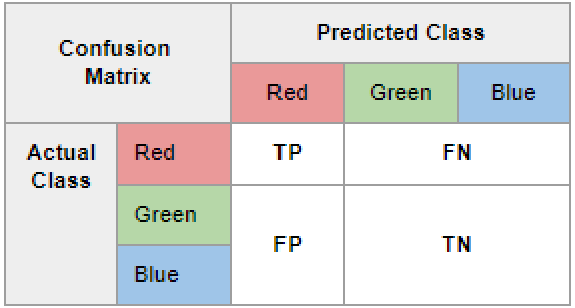
Model Performance Metrics

Confusion Metrics Example

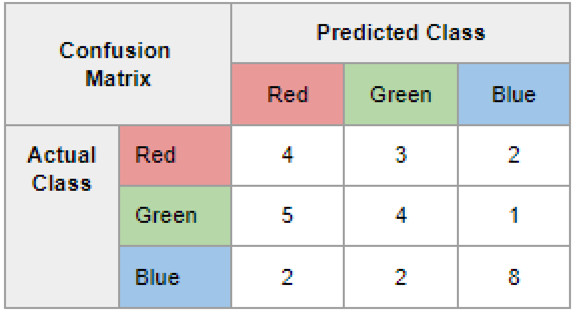


Confusion Matrix		Predicted Class		
		Red	Green	Blue
Actual Class	Red	4	3	2
	Green	5	4	1
	Blue	2	2	8

Positive class = “Red”

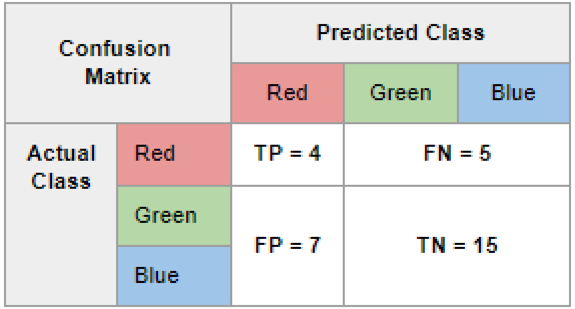


Confusion Matrix		Predicted Class		
		Red	Green	Blue
Actual Class	Red	TP	FN	
	Green	FP	TN	
	Blue			



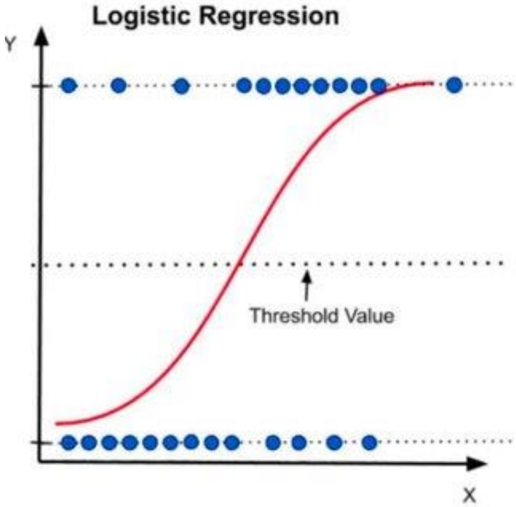
Confusion Matrix		Predicted Class		
		Red	Green	Blue
Actual Class	Red	4	3	2
	Green	5	4	1
	Blue	2	2	8

Positive class = “Red”

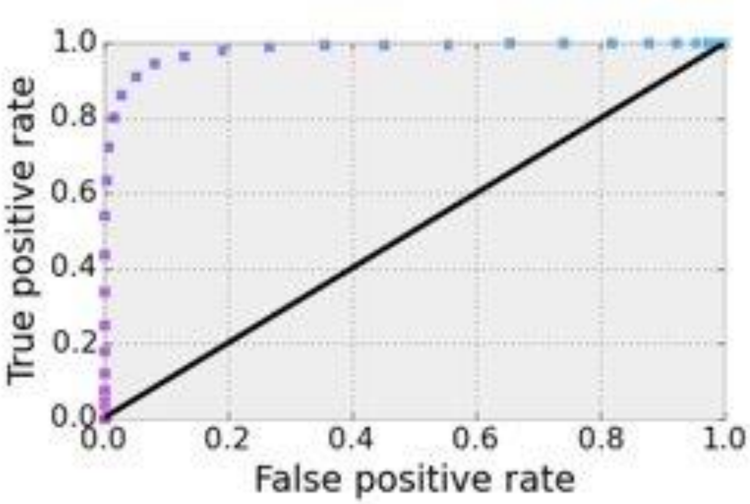


Confusion Matrix		Predicted Class		
		Red	Green	Blue
Actual Class	Red	TP = 4	FN = 5	
	Green	FP = 7	TN = 15	
	Blue			

ROC Curves

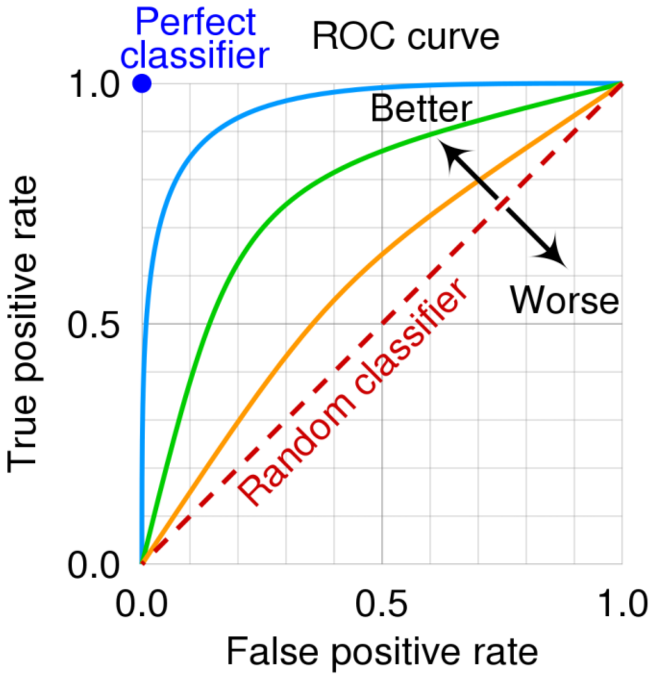


Logistic Regression
Y
Threshold Value
X



1.0
0.8
0.6
0.4
0.2
0.0
True positive rate
0.0
0.2
0.4
0.6
0.8
1.0
False positive rate

Receiver operating Characters



ROC plot shows the true positive rate as a function of the false positive rate.

True Positive Rate / Recall / Sensitivity
TPR = TP / (TP + FN).

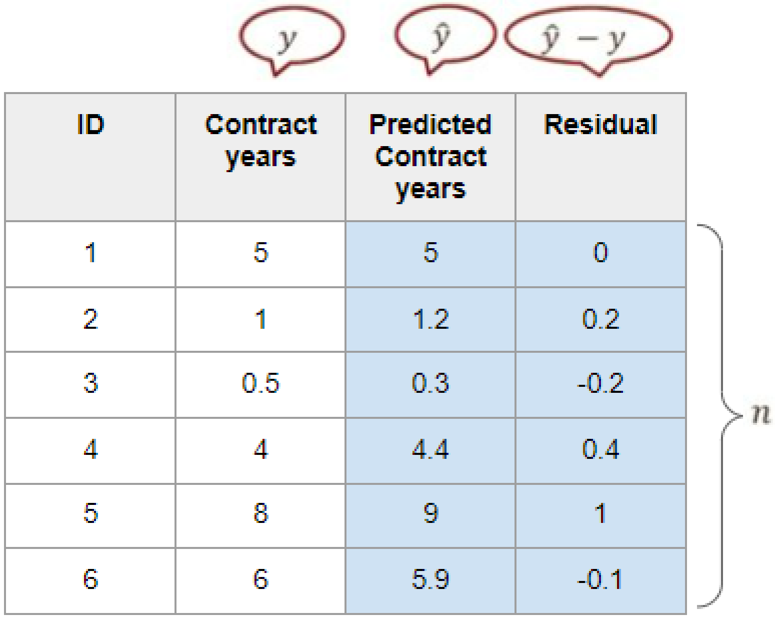
False positive rate / Specificity
FPR = FP / (FP + TN).

The diagonal corresponds to the random guess.

AUC = Area under (ROC) Curve: 1 = best value, 0.5 = random guess.

Experiment with different thresholds to balance the trade-off.

Regression Performance Metrics


y
$\hat{y}$
$\hat{y} - y$
n

ID	Contract years	Predicted Contract years	Residual
1	5	5	0
2	1	1.2	0.2
3	0.5	0.3	-0.2
4	4	4.4	0.4
5	8	9	1
6	6	5.9	-0.1

- **Mean absolute error (MAE)**: Large errors have the same power as small errors

$$\frac{1}{n} \cdot \Sigma_n |\hat{y} - y| \quad = \mathbf{0.32}$$

- **Root-mean-square error (RMSE)**: Assigns more weight to large errors

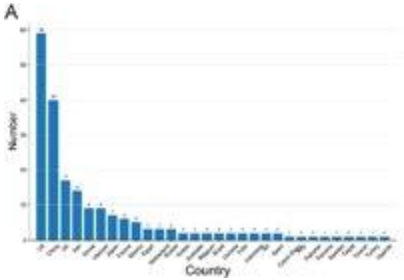
$$\sqrt{\frac{1}{n} \cdot \Sigma_n (\hat{y} - y)^2} \quad = \mathbf{0.46}$$

- **R²**: How much variability in Y can be explained by the model?

$$R^2 = 1 - \frac{RSS}{TSS}$$

Imbalanced Data

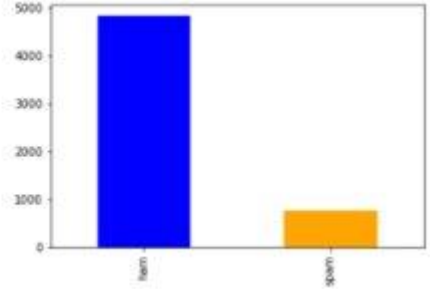
Imbalanced Datasets



Class imbalance in features

Examples:

- Customers from different countries
- Sales over different products
- Customer feedback in different languages



Class imbalance in targets

Examples:

- Fraud detection
- Spam filtering
- Subscription churn
- Advertising clicks

Feature Variable Strategies

Refactoring: Merge minority classes into one joint class.

Example: Keep top 10 countries, collect everything else under “Other”.

Segmenting: Separate the data into different subsets and train different (less complex) model for the minority classes.

Example: Different model for each continent or sales region.

Target Variable Strategies

Oversampling the minority class

- Randomly duplicate samples from the minority class until the classes are balanced out.

Undersampling the majority class

- Remove samples from the majority class randomly until both classes contain an equal number of observations.

Generate synthetic data

- Create new, synthetic samples for the minority class by interpolating existing data points.

Penalize algorithms

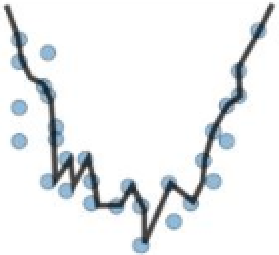
- Modify learning algorithm to consider class imbalance.

Try a different algorithm

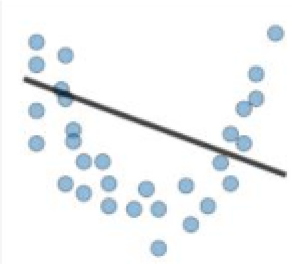
- Algorithms that usually work well on imbalanced data: random forests, boosted trees, naive bayes classifiers, K nearest neighbors.
- Algorithms that usually do not work well on imbalanced data: linear regression, logistic regression, neural networks.

Overfitting and Cross-Validation

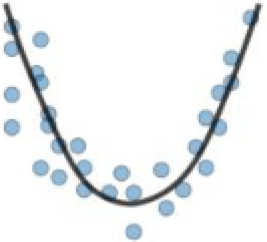
Overfitting



Overfit

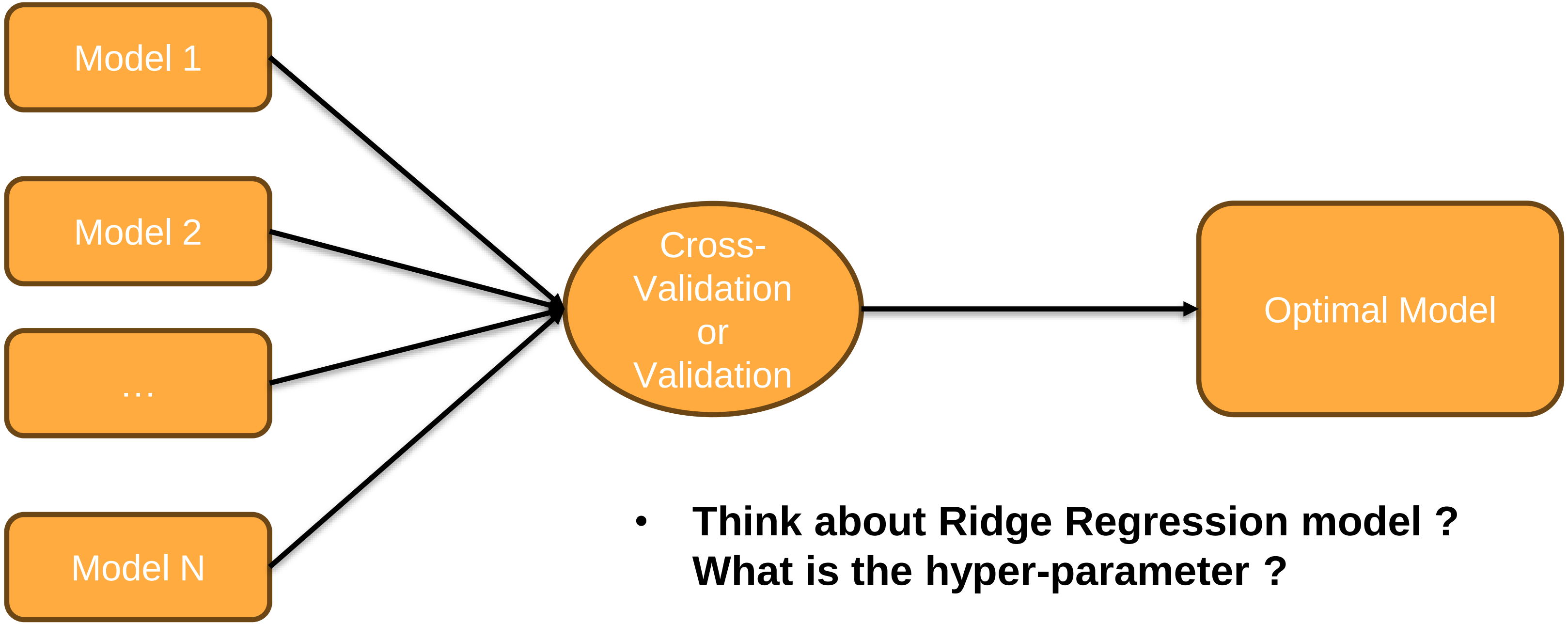


Good fit



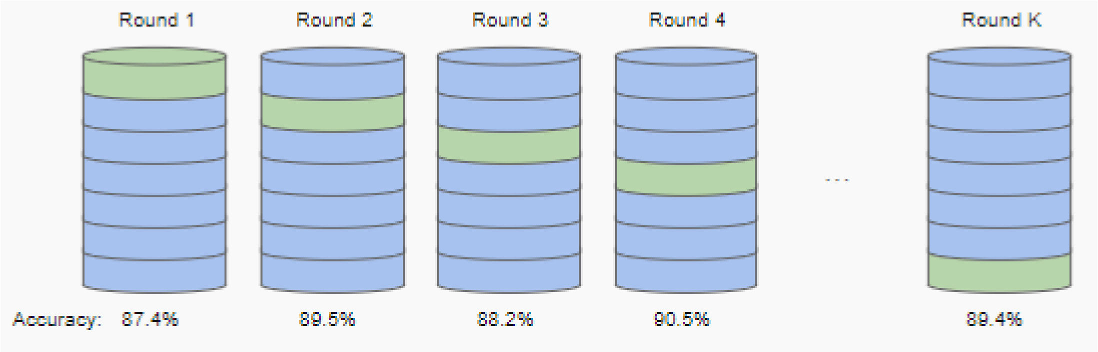
Underfit

Model Selection



- **Think about Ridge Regression model ? What is the hyper-parameter ?**
- **Think about Polynomial Regression model ? What is the hyper-parameter ?**

Cross Validation – For Model Selection/Hyper-parameter Tuning

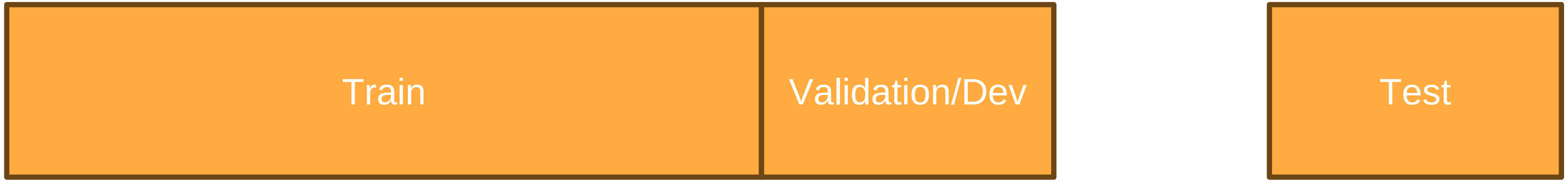


Split the data into K folds.

K-1 folds are used for training, 1-fold for validation.

Shuffle the validation fold k times.

Alternatively



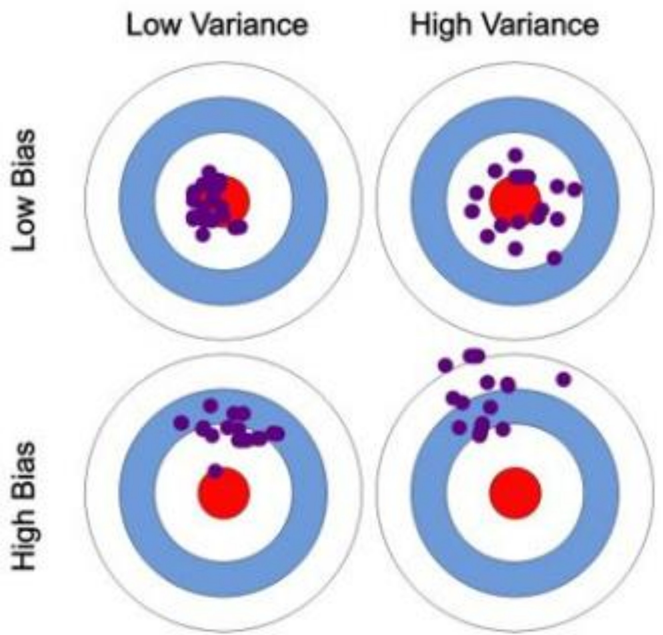
Train

Validation/Dev

Test

Bias and Variance Trade-Offs

Source of Errors



Bias:

- The truth is different from what the technique can capture. Since the truth is unknown, one must make assumptions.

Variance

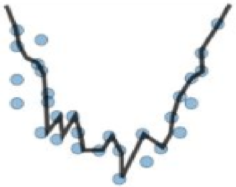
- Since the algorithm has only finite data, it cannot find the optimal model.

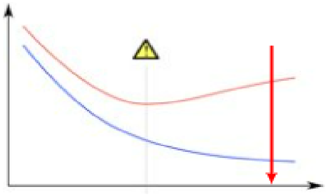
Noise

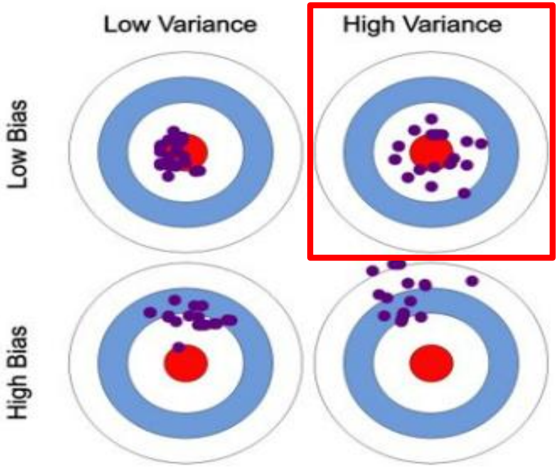
- Data contains random noise which cannot (and should not) be captured by the model.

Overfit

This model has **high variance and low bias**. It is **overfitting** the training data and will very likely show a large error on new data.

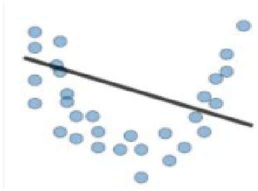


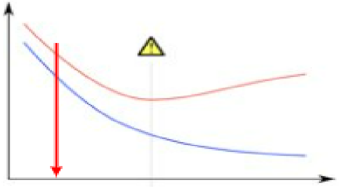


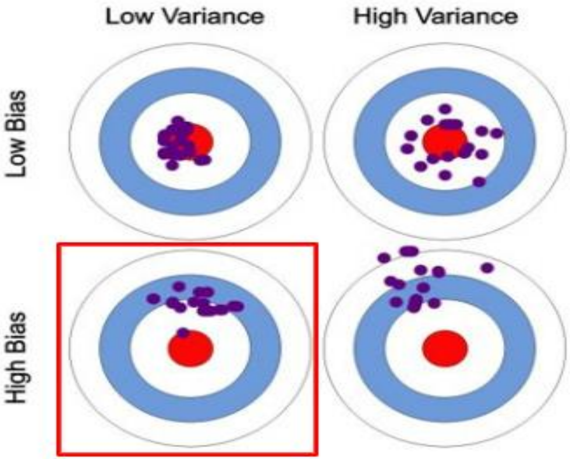


Underfit

This model has **low variance and high bias**. It is **underfitting** the training data and has a large error on training and validation data.

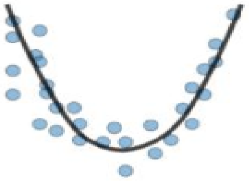


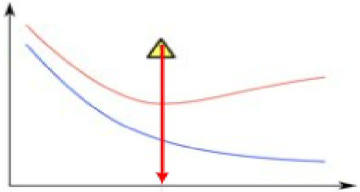


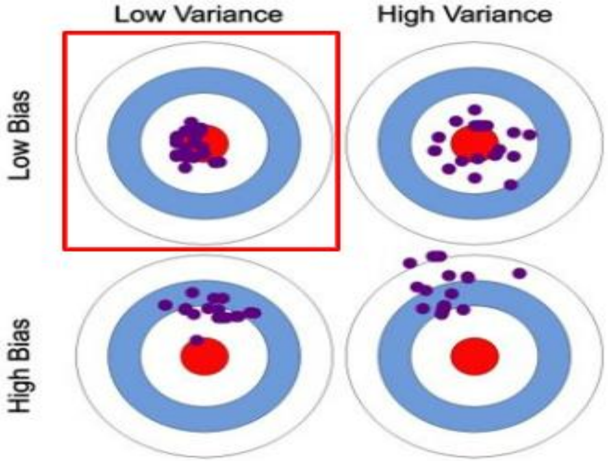


Good fit

This model offers a **good trade-off between variance and bias**. It shows a **good fit for the data.**

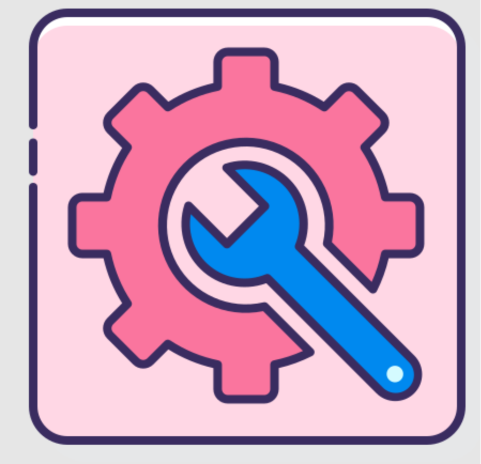






Hyperparameter ang Grid Search

Hyperparameter



Hyperparameters control the machine learning modeling process, controlling the way our algorithm runs and how it converges on a solution.

Hyperparameters help the ML process to find the **parameters** we are interested in (e.g., Beta coefficients in regression).

Example: Elastic Net Regression

$$L_{enet}(\hat{\beta}) = \frac{\sum_{i=1}^{n}(y_i - x_i^T\hat{\beta})^2}{2n} + \lambda \underset{\text{Ridge}}{\boxed{\Bigg(\frac{1-\alpha}{2}\sum_{j=1}^{m}\hat{\beta}_j{}^2}} + \underset{\text{Lasso}}{\boxed{\alpha\sum_{j=1}^{m}|\hat{\beta}_j|\Bigg)}}$$

- α and λ are hyper-parameters
- β are model parameters (determined by closed-form solution or gradient descent)

Hyper-parameters combinations

α\λ	0.1	0.2	0.3
0.1	0.1,0.1	…	…
0.2	…	…	
0.3	…	…	0.3,0.3

Grid Search

Grid search is a method of trying out different values for hyperparameters to find the ones that result in best model performance.

Brute force method: Trying all possible combinations of hyperparameter values that we specify.

This typically involves creating a grid of all possible combinations of the hyperparameter values.

Grid search can be very time-consuming, especially when trying many hyperparameters and/or a large range of values over large amounts of data and/or complex models.

Tips for Effective Grid Search

Use a validation set for model evaluation, not just the training set (ideally cross validation).

If using cross validation, use the same cross validation scheme throughout the grid search.

Be aware grid search can be very time-consuming.

Start grid search with a small number of hyperparameters and narrow down the search space.

Don't forget to retrain your model on all training data (train + dev) with the best hyperparameters combination found.

Cross Validation and Grid Search

Thank you