**Table Of Contents**

[Add more data](#_5g05lkr0g0iz) 1

[Treat missing and outlier values](#_qom7ifqbgfqi) 1

[Feature Engineering](#_hn1a0be3lxv6) 2

[Feature Selection](#_yhms5ixukiv3) 2

[Algorithm Tuning](#_grlfnhg0v3u2) 2

[Ensemble Methods](#_edg6mgb2181v) 3

[Cross Validation](#_tyczkrtdjajx) 3

# Add more data

* The more data you have, the better models are able to categorize and discover trends in the data
* However, if there is too much data, the models may be prone to overfitting to the testing data. Thus, applying any of the other methods in this document will help.

# Treat missing and outlier values

* Often, missing data or outliers reduce the accuracy of the model or may result in bias. Thus, one should often treat missing and outlier values before performing any sort of data analysis of machine learning.
* To deal with missing values or outliers, you can…
* **Missing Values:**
  + In the case of continuous (numerical) values, you can impute the missing values with the mean, median, or mode of the dataset. For categorical values, you can treat the variables as a separate class. You can also build a model to predict those categorical values.
  + **KNN Imputation** is also an alternative, which works by finding the k-nearest neighbours to the missing data point and imputing them based on the non-missing values in the neighbors
* **Outliers:**
  + Delete observations, perform transformation, binning, imputation, or treat outlier variables separately

# Feature Engineering

* Extracts more information from existing data
  + New information is extracted in terms of new features
  + These features may have a greater ability to explain variance in the testing set
* Highly influenced by hypothesis generation; requires a specific, detailed hypothesis.
* Divided into two steps:
  + **Feature Transformation:** May not have the same interpretation as the original features, but may have more discriminatory powers. Examples include:
    - Normalizing features to a range between 0 and 1
    - Principal Component Analysis
    - Random Projection
    - Neural Networks
    - Transforming categorical features to numerical
  + **Feature Creation:** Creating new features from existing features. Example: finding transactions/day from transactions and day variables, which may be more important to the model than the original two variables separately.

# Feature Selection

* The process of finding the best subset of attributes which better explains the relationship of independent variables with target variable
* You can select the useful features based on existing metrics like
  + **Domain Knowledge**: Based on domain experience, select features which may have a higher impact on target variable
  + **Visualization:** Use matplotlib or any visual library to group the categories, which made variable selection easier
  + **Statistical Parameters:** Consider p-values, information values, and other statistical metrics to select right features
    - Can use Principal Component Analysis

# Algorithm Tuning

* Best place to better results is to start with your models that already perform the best, as these will likely be the best performers even if other models are also tuned.
* Machine learning algorithms are parameterized and modification of those parameters can lead to more accurate models. Below are parameters of the more common models provided in the scikit-learn library in Python
  + **Linear Regression**: fit\_intercept, normalize, copy\_X, n\_jobs
  + **Logistic Regression:** penalty, dual, tol, C, fit\_intercept, intercept\_scaling, class\_weight, random\_state, solver, max\_iter, multi\_class, verbose, warm\_start, n\_jobs, l1\_ratio
  + **Decision Tree Classifier**: criterion, splitter, max\_depth, min\_samples\_split, min\_samples\_leaf, min\_weight\_fraction\_leaf, max\_features, random\_state, max\_leaf\_nodes, min\_impurity\_decrease, class\_weight, ccp\_alpha

# Ensemble Methods

* Involves combining the results of multiple weaker models to get improved results
* Works the best when you have multiple optimized models that each specialize in different sectors
* Can be achieved in multiple ways:
  + **Bagging** (Bootstrapped Aggregation): The same algorithm has different perspectives on the model by being trained on different subsets of the data (commonly achieved via random sampling, in the case of RF)
  + **Boosting:** Different algorithms are trained on the same training data
  + **Blending** (or stacking): Variety of models whose predictions are taken as input to a new model that learns how to combine the predictions into an overall prediction
    - Differs from the other two in that blending considers heterogeneous weak learners, compared to bagging and boosting, which consider homogeneous weak learners
* Often extremely effective and often used by winners of Kaggle competitions

# Cross Validation

**Validation**: The process of deciding whether the numerical values quantifying hypothesized relationships between variables are acceptable as descriptions of the data

* Generally, an error estimation of the data is made after training, better known as evaluation of residuals
* In this process, a numerical estimate of the difference in predicted and original responses is done, and is known as the training error
* However, this only measures the training set, and doesn’t reveal if underfitting or overfitting is taking place. The process of generalizing the training to the test data set is called **cross validation**

**Method**

* Involves removing a part of the training set and using it to get predictions from the model trained on the rest of the data (similar to the train/test split model in the first place)
* This is a simple kind of cross validation known as the **holdout method**
  + Still suffers from issues of high variance
  + This is because the model is not sure which data points will end up in the evaluation set and the result may be different entirely for different sets

**K-Fold Cross Validation**

* In regular cross validation, we remove a part of the data as the testing set and use the rest of the data to draw predictions. However, removing a part of the data set is always risky and may result in increased error induced by bias
* Solution: K-Fold Cross Validation
  + The data is divided into k subsets (up to user). The holdout method is repeated k times such that each time, one of the k subsets is used as the test/holdout set and the other k-1 sets are put together to form a training set.
  + The error estimation is averaged over all k trials to get total effectiveness of the model
  + This significantly reduces bias as we are using most of the data for fitting, and also reduces variance since most of the data is being used in the validation set

Further K-Fold Cross Validation variants:  
**Stratified K-Fold Cross Validation**

**Leave P-Out Cross Validation**

**Exhaustive Cross Validation**

**Non-Exhaustive Cross Validation**

# 