Baderia Global Institute of Engineering and Management

Department of Computer Science and Engineering

GLOBAL
Group of Institutions
The Best

Name of the Student:	____________________
Enrolment Number:	____________________
Semester:	**VI**
Department:	**Computer Science and Engineering**
Subject:	**Machine Learning Lab Manual**

Machine Learning Lab Manual

Contents

Vision and Mission of the Institute

Vision of Institute

Transforming life by providing professional education with excellence.

Mission of Institute

1. Quality Education: Providing Education with quality and shaping up Technocrats and budding managers with a focus on adapting to changing technologies.
2. Focused Research & Innovation: Focusing on Research and Development and fostering Innovation among the academic community of the Institution.
3. People Focused: Accountable and committed to institutional operations for effective functioning by Faculty members, Staff and Students.
4. Holistic Learning: Focus on conceptual learning with practical experience and experiential learning with strong Industrial connections and collaborations.
5. Service to Society: Providing Technical and Managerial services to society for betterment of their quality of life with best of the skills, compassion and empathy.

Vision and Mission of the Department

Vision of the Department

Transforming the life of the graduates by providing excellent education in the field of Computer Science & Engineering.

Mission of the Department

The department strives:

1. Create student centric learning ambience so as to produce graduates who are well informed about latest technological trends and advancement in the world of computing, technology and research.
2. Produce professionals who are capable to work in diversified fields, find workable solutions to complex problems with awareness and concern for society and environments.
3. Continuously upgrade faculty through training so that they function effectively.
4. Encourage industry institute collaborations through consultancies and research, helping students to have conceptual learning.

Program Outcomes

1. Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization for the solution of complex engineering problems.

2. Problem analysis: Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for public health and safety, and cultural, societal, and environmental considerations.

4. Conduct investigations of complex problems: Use research based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of information to provide valid conclusions.

5. Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools, including prediction and modeling to complex engineering activities, with an understanding of the limitations.

6. The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. Communication: Communicate effectively on complex engineering activities with the engineering community and with the society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Course Outcomes

After Completing the course student should be able to:

CO1. Apply knowledge of computing and mathematics to machine learning problems, models and algorithms.

CO2. Analyze a problem and identify the computing requirements appropriate for its solution.

CO3. Apply mathematical foundations, algorithmic principles, and computer science theory to the modelling and design of computer-based systems in a way that demonstrates comprehension of the trade-offs involved in design choices.

LABORATORY REGULATIONS AND SAFETY RULES

The following Regulations and Safety Rules must be observed in all concerned laboratory locations.

1. It is the duty of all concerned parties who use any electrical laboratory to take all reasonable steps to safeguard the HEALTH and SAFETY of themselves and all other users and visitors.
2. Make sure that all equipment is properly working before using them for laboratory exercises. Any defective equipment must be reported immediately to the Lab. Instructors or Lab. Technical Staff.
3. Students are allowed to use only the equipment provided in the experiment manual or equipment used for senior project laboratory.
4. Power supply terminals connected to any circuit are only energized with the presence of the Instructor or Lab. Staff.
5. Students should keep a safe distance from the circuit breakers, electric circuits or any moving parts during the experiment.
6. Avoid any part of your body to be connected to the energized circuit and ground.
7. Switch off the equipment and disconnect the power supplies from the circuit before leaving the laboratory.
8. Observe cleanliness and proper laboratory housekeeping of the equipment and other related accessories.
9. Wear proper clothes and safety gloves or goggles required in working areas that involves fabrications of printed circuit boards, chemicals process control system, antenna communication equipment and laser facility laboratories.
10. Double check your circuit connections specifically in handling electrical power machines, AC motors and generators before switching "ON" the power supply.
11. Make sure that the last connection to be made in your circuit is the power supply and first thing to be disconnected is also the power supply.
12. Equipment should not be removed, transferred to any location without permission from the laboratory staff.
13. Software installation in any computer laboratory is not allowed without the permission from the Laboratory Staff.
14. Computer games are strictly prohibited in the computer laboratory.
15. Students are not allowed to use any equipment without proper orientation and actual hands on equipment operation.

List of Experiments

S. No.	Experiment	Date of Completion	Date of Checking	Sign
1.	Implement a Gaussian Probability Distribution Function (PDF) and Gaussian Cumulative Distribution Function (CDF) with a sample space of -5 to 5, mean zero and standard deviation 1. Use scipy module.			
2.	Implement a Chi- Squared Probability Distribution Function (PDF) and Chi- Squared Cumulative Distribution Function (CDF) with a sample space of 0 to 50, step size 0.01 and degree of freedom is 20. Use scipy module.			
3.	Implement a ML dataset having class imbalance using pandas and scikit-learn. Demonstrate how the class proportion can be preserved while splitting irrespective of the class imbalance?			
4.	Using a ML dataset in pandas and scikit-learn show that if cross-validation is used and the samples are not in an arbitrary order, shuffling may be required to get meaningful results.			
5.	Using a ML dataset in pandas and scikit-learn demonstrate pipeline slicing showing operation on some part of pipeline instead of the whole pipeline.			
6.	Create multiple models and ensemble them using VotingClassifier to improve the classifier's accuracy.			
7.	With a tree-based model having nominal (unordered) features implement OrdinalEncoder and demonstrate that OrdinalEncoder is faster than the OneHotEncoder and accuracy is also the same.			
8.	Implement feature encoders to encode categorical features: ● OneHotEncoder for unordered (nominal) data ● OrdinalEncoder for ordered (ordinal) data			
9.	Implement feature selection to a Pipeline: 1. Use SelectPercentile to keep the highest scoring features 2. Add feature selection after preprocessing but before model building			

Experiment 01

Aim- Implement a Gaussian Probability Density Function (PDF) and Gaussian Cumulative Distribution Function (CDF) with a sample space of -5 to 5, mean zero and standard deviation 1. Use scipy module.

Platform- Google Colaboratory

Theory-

Probability Density Functions

A **probability density function** (pdf) tells us the probability that a random variable takes on a certain value.

Cumulative Distribution Functions

A **cumulative distribution function** (cdf) tells us the probability that a random variable takes on a value less than or equal to *x*.

Gaussian distribution

Normal distribution, also known as the Gaussian distribution, is **a probability distribution that is symmetric about the mean, showing that data near the mean are more frequent in occurrence than data far from the mean**. In graph form, normal distribution will appear as a bell curve. There are several parts of machine learning that takes advantage from using a Gaussian distribution. Those areas are including;

- Gaussian processes
- Variational inference
- Reinforcement learning

It is similarly broadly used in other application areas such as

- Signal processing such as Kalman filter
- Control, for example linear quadratic regulator
- Statistics, as hypothesis testing

Practical uses

- Designing Standardized Tests: These are designed as a result that test-taker scores fall within a Gaussian distribution.
- Statistical Tests: There may be several statistical Tests are derived from a Gaussian distribution.
- Quantum Mechanics: The Gaussian distribution may be used to define the ground state of a quantum harmonic oscillator.

Importance of Gaussian distribution

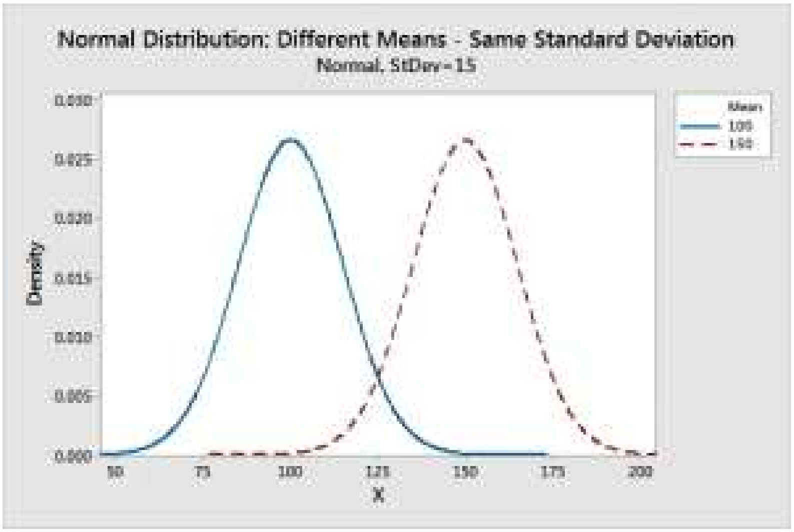
- It is ever-present as a dataset with finite variance turns into Gaussian as long as dataset with free feature-probabilities is permitted to raise in size.
- It is the most significant probability distribution in statistics as it turns many natural phenomena such as age, height, test-scores, IQ scores, and sum of the rolls of two cubes and so on.
- Datasets with Gaussian distributions creates valid to a diversity of methods that decrease under parametric statistics.
- The approaches for example propagation of uncertainty and least squares parameter right are related only to datasets with normal or normal-like distributions.
- Reviews and conclusions resulting from such analysis are intuitive. That also easy to explain to audiences with basic knowledge of statistics.

Parameters of Gaussian distribution

The mean and standard deviation are two main parameters of a Gaussian distribution. We may decide the shape and probabilities of the distribution with respect to our problem statement by the help of these parameters. The shape of the distribution changes when the parameter value changes.

Mean

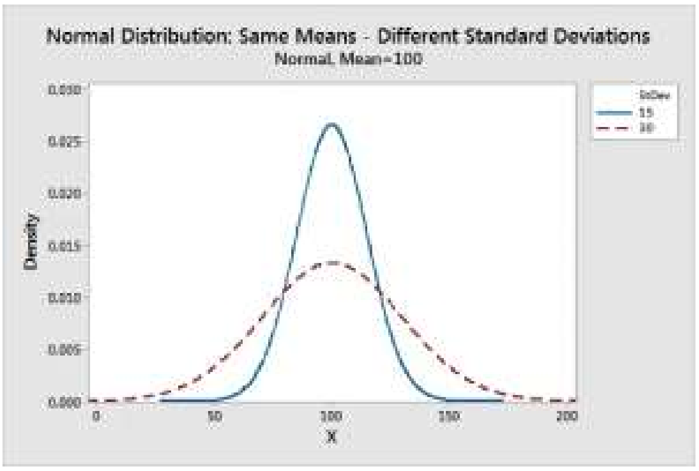
- Scientists used the mean or average value as a measure of dominant trend.
- It may be used to define the distribution of variables that are measured as ratios or intervals.
- The mean decides the site of the peak. The many of the data points are gathered around the mean in a normal distribution graph.
- By changing the value of the mean we can move the curve of Gaussian distribution moreover to the left or right along the X-axis.



Standard Deviation

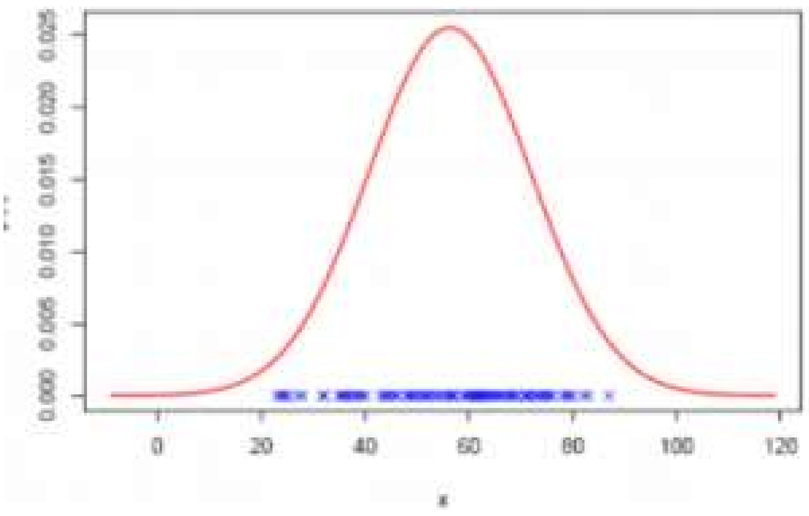
- It acts how the data points are circulated comparative to the mean.
- It fixes how distant the data points are missing from the mean.

- It characterizes the distance between the mean and the data points.
- It describes the width of the graph. Consequently, varying the value of standard deviation make tighter or enlarges the width of the distribution along the x-axis.
- A lesser standard deviation with respect to the mean results in a steep curve and a greater standard deviation marks in a flatter curve.



- When a single variable x, the Gaussian distribution may be written in the form:

$$\mathcal{N}(x|\mu,\sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\}$$



- where μ is the mean and σ2 is the variance

- For a D-dimensional vector x, the multivariate Gaussian distribution can be written in the form:

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \exp\left\{ -\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} (\mathbf{x}-\boldsymbol{\mu}) \right\}$$

- where μ is a D-dimensional mean vector
- Σ is a D ×D covariance matrix
- |Σ| denotes the determinant of Σ

Result- The result has been verified under experimental error conditions.

Experiment 02

Aim- Implement a Chi- Squared Probability Density Function (PDF) and Chi- Squared Cumulative Distribution Function (CDF) with a sample space of 0 to 50, step size 0.01 and degree of freedom is 20. Use scipy module.

Platform- Google Colaboratory

Theory-

The Chi-Square test is a statistical procedure for determining the difference between observed and expected data. This test can also be used to determine whether it correlates to the categorical variables in our data. It helps to find out whether a difference between two categorical variables is due to chance or a relationship between them.

Formula for Chi-Square Test

$$x_c^2 = \frac{\Sigma \, (O_i \; - E_i)^2}{E_i}$$

Where

c = Degrees of freedom

O = Observed Value

E = Expected Value

The degrees of freedom in a statistical calculation represent the number of variables that can vary in a calculation. The degrees of freedom can be calculated to ensure that chi-square tests are statistically valid. These tests are frequently used to compare observed data with data that would be expected to be obtained if a particular hypothesis were true.

The Observed values are those you gather yourselves.

The expected values are the frequencies expected, based on the null hypothesis.

Why Do You Use the Chi-Square Test?

Chi-square is a statistical test that examines the differences between categorical variables from a random sample in order to determine whether the expected and observed results are well-fitting.

Here are some of the uses of the Chi-Squared test:

- The Chi-squared test can be used to see if your data follows a well-known theoretical probability distribution like the Normal or Poisson distribution.
- The Chi-squared test allows you to assess your trained regression model's goodness of fit on the training, validation, and test data sets.

Limitations of Chi-Square Test

There are two limitations to using the chi-square test that you should be aware of.

- The chi-square test, for starters, is extremely sensitive to sample size. Even insignificant relationships can appear statistically significant when a large enough sample is used. Keep in mind that "statistically significant" does not always imply "meaningful" when using the chi-square test.
- Be mindful that the chi-square can only determine whether two variables are related. It does not necessarily follow that one variable has a causal relationship with the other. It would require a more detailed analysis to establish causality.

Result- The result has been verified under experimental error conditions.

Experiment 03

Aim- Implement a ML dataset having class imbalance using pandas and scikit-learn. Demonstrate how the class proportion can be preserved while splitting irrespective of the class imbalance?

Platform- Google Colaboratory

Theory-

Class imbalance is a scenario where the number of observations belonging to one class is significantly lower than those belonging to the other classes.

This problem is predominant in scenarios where anomaly detection is crucial like electricity pilferage, fraudulent transactions in banks, identification of rare diseases, etc. In this situation, the predictive model developed using conventional machine learning algorithms could be biased and inaccurate.

This happens because Machine Learning Algorithms are usually designed to improve accuracy by reducing the error. Thus, they do not take into account the class distribution / proportion or balance of classes.

Challenges with standard Machine learning techniques

The conventional model evaluation methods do not accurately measure model performance when faced with imbalanced datasets.

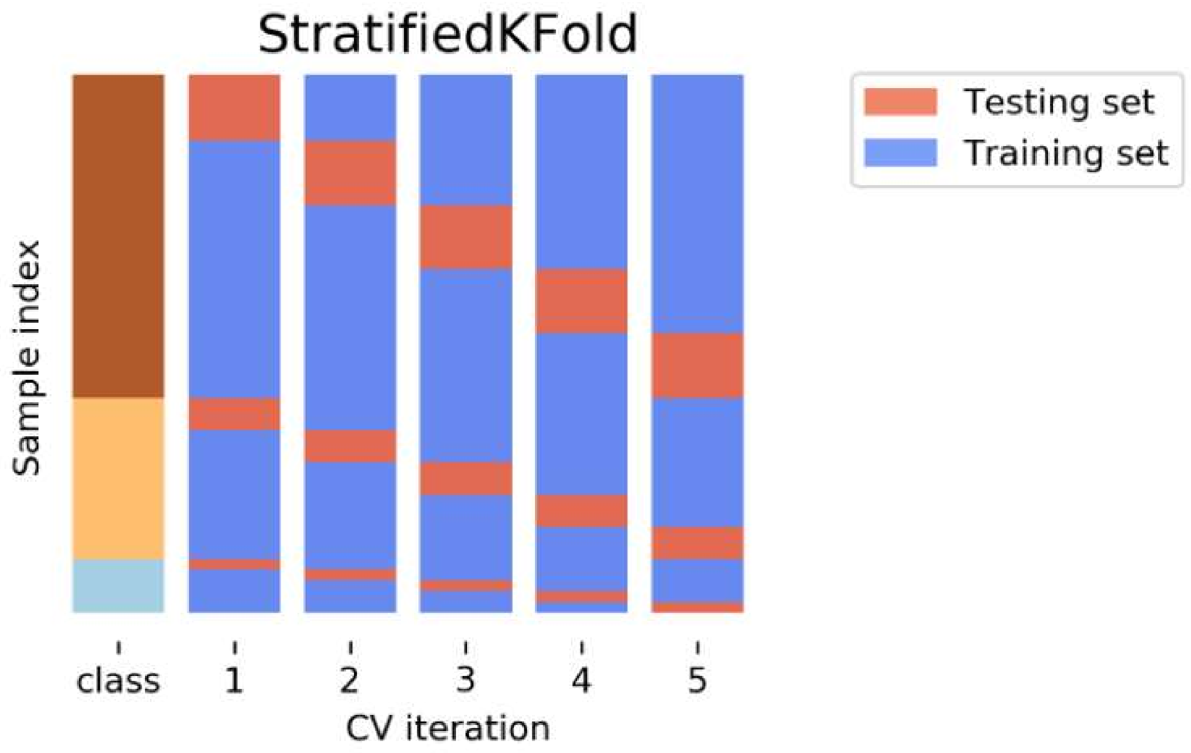
Standard classifier algorithms like Decision Tree and Logistic Regression have a bias towards classes which have number of instances. They tend to only predict the majority class data. The features of the minority class are treated as noise and are often ignored. Thus, there is a high probability of misclassification of the minority class as compared to the majority class.

Evaluation of a classification algorithm performance is measured by the Confusion Matrix which contains information about the actual and the predicted class.

When we change the random_state inside the train_test_split class, the model's accuracy is changed, and this will be changing to any value of random_state due to the fact that we cannot observe the right accuracy of the model. As its name justifies, it samples the data without taking care of the distribution of classes. Say you are dealing with binary classification out of 100% dataset 70% belong to class 0 and remaining to class 1; for this balance, if you do random sampling, there is a high chance of getting different class distributions between training and testing. By tearing on such a dataset, you will get poor accuracy.

The most used validation technique is K-Fold Cross-validation which involves splitting the training dataset into k folds. The first k-1 folds are used for training, and the remaining fold is held for testing, which is repeated for K-folds. A total of K folds are fit and evaluated, and the mean accuracy for all these folds is returned. This process has shown an optimistic result for balanced classification tasks, but it fails for imbalance classes. This is due to cross-validation, which also splits the data randomly without taking care of the class imbalance.

So the solution is not to split the data randomly, but it should be split in a stratified manner. The stratified k fold cross-validation is an extension of the cross-validation technique used for classification problems.



It maintains the same class ratio throughout the K folds as the ratio in the original dataset. So, for example, you are dealing with diabetes prediction in which you have the class ratio of 70/30; by using stratified K fold, the same class ratio is preserved throughout the K folds.

Result- The result has been verified under experimental error conditions.

Experiment 04

Aim- Using a ML dataset in pandas and scikit-learn show that if cross-validation is used and the samples are not in an arbitrary order, shuffling may be required to get meaningful results.

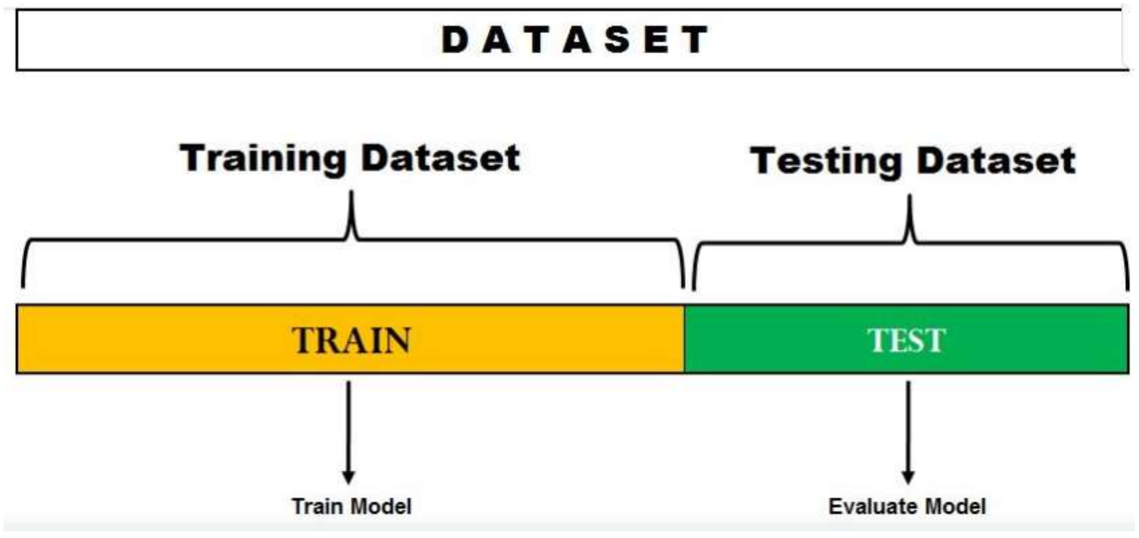
Platform- Google Colaboratory

Theory-

Cross-Validation is a resampling technique with the fundamental idea of splitting the dataset into 2 parts- training data and test data. Train data is used to train the model and the unseen test data is used for prediction. If the model performs well over the test data and gives good accuracy, it means the model hasn't over fitted the training data and can be used for prediction.

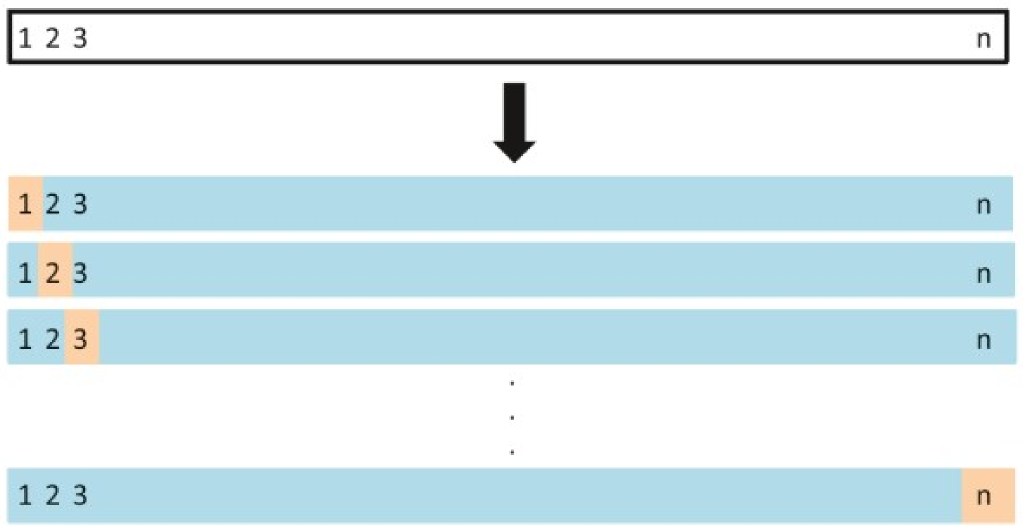
1. Hold Out method

This is the simplest evaluation method and is widely used in Machine Learning projects. Here the entire dataset (population) is divided into 2 sets – train set and test set. The data can be divided into 70-30 or 60-40, 75-25 or 80-20, or even 50-50 depending on the use case. As a rule, the proportion of training data has to be larger than the test data.



2. Leave One out Cross-Validation

In this method, we divide the data into train and test sets – but with a twist. Instead of dividing the data into 2 subsets, we select a single observation as test data, and everything else is labelled as training data and the model is trained. Now the 2nd observation is selected as test data and the model is trained on the remaining data.



This process continues 'n' times and the average of all these iterations is calculated and estimated as the test set error.

$$\mathrm{CV}_{(n)} = \frac{1}{n}\sum_{i=1}^{n}\mathrm{MSE}_i.$$

When it comes to test-error estimates, LOOCV gives unbiased estimates (**low bias**). But bias is not the only matter of concern in estimation problems. We should also consider variance.

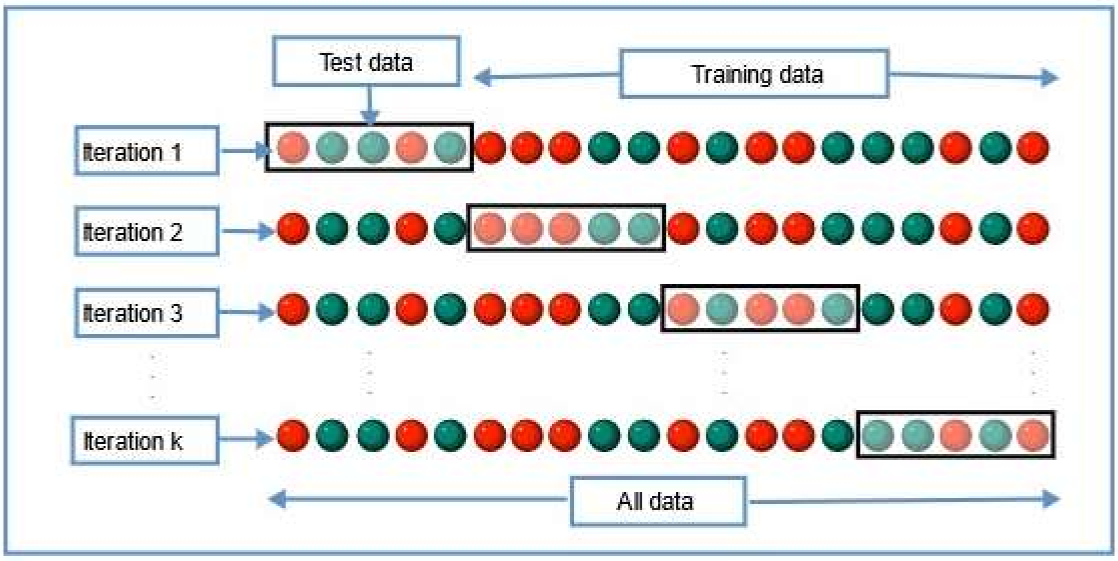
LOOCV has an extremely **high variance** because we are averaging the output of n-models which are fitted on an almost identical set of observations, and their outputs are highly positively correlated with each other.

And you can clearly see this is computationally expensive as the model is run 'n' times to test every observation in the data.

3. K-Fold Cross-Validation

In this resampling technique, the whole data is divided into k sets of almost equal sizes. The first set is selected as the test set and the model is trained on the remaining k-1 sets. The test error rate is then calculated after fitting the model to the test data.

In the second iteration, the 2nd set is selected as a test set and the remaining k-1 sets are used to train the data and the error is calculated. This process continues for all the k sets.



The mean of errors from all the iterations is calculated as the CV test error estimate.

$$\mathrm{CV}_{(k)} = \frac{1}{k}\sum_{i=1}^{k} \mathrm{MSE}_i .$$

In K-Fold CV, the no of folds k is less than the number of observations in the data (k<n) and we are averaging the outputs of k fitted models that are somewhat less correlated with each other since the overlap between the training sets in each model is smaller. This leads to **low variance** then LOOCV.

The best part about this method is each data point gets to be in the test set exactly once and gets to be part of the training set k-1 times. As the number of folds k increases, the variance also decreases (low variance). This method leads to **intermediate bias** because each training set contains fewer observations (k-1) n/k than the Leave One Out method but more than the Hold Out method.

Typically, K-fold Cross Validation is performed using k=5 or k=10 as these values have been empirically shown to yield test error estimates that neither have high bias nor high variance.

The major disadvantage of this method is that the model has to be run from scratch k-times and is computationally expensive than the Hold Out method but better than the Leave One Out method.

Result- The result has been verified under experimental error conditions.

Experiment 05

Aim- Using a ML dataset in pandas and scikit-learn demonstrate pipeline slicing showing operation on some part of pipeline instead of the whole pipeline.

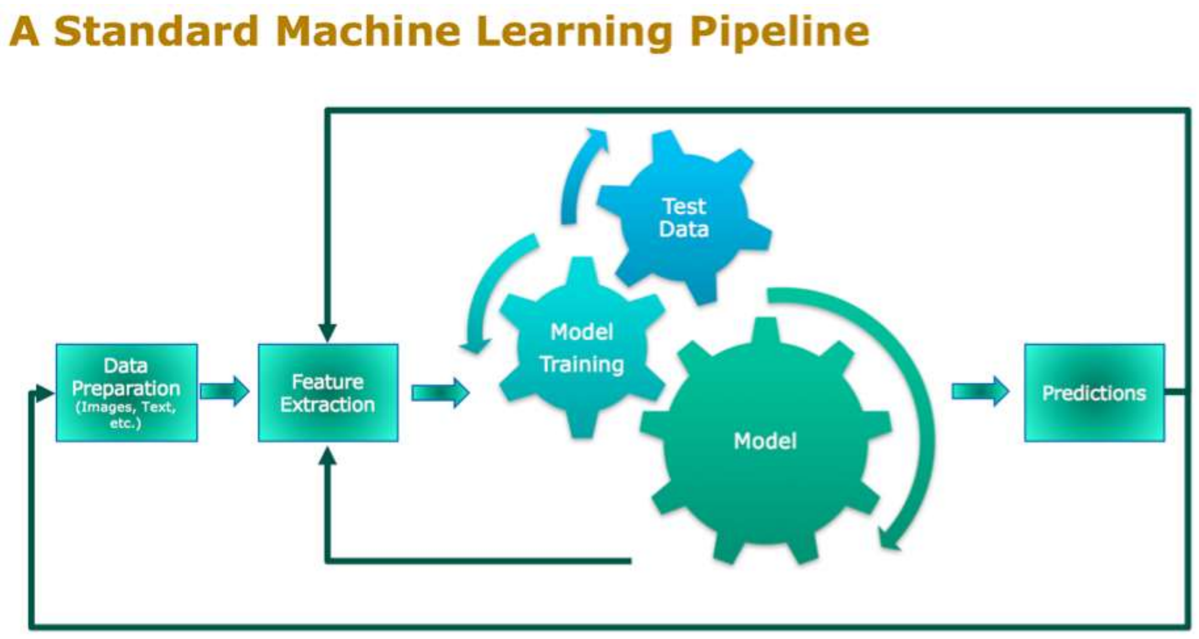
Platform- Google Colaboratory

Theory-

For building any machine learning model, it is important to have a sufficient amount of data to train the model. The data is often collected from various resources and might be available in different formats. Due to this reason, data cleaning and preprocessing become a crucial step in the machine learning project.

Whenever new data points are added to the existing data, we need to perform the same preprocessing steps again before we can use the machine learning model to make predictions. This becomes a tedious and time-consuming process!

An alternate to this is creating a machine learning pipeline that remembers the complete set of preprocessing steps in the exact same order. So that whenever any new data point is introduced, the machine learning pipeline performs the steps as defined and uses the machine learning model to predict the target variable.


A Standard Machine Learning Pipeline
Test Data
Model Training
Model
Data Preparation (Images, Text, etc.)
Feature Extraction
Predictions

A machine learning pipeline is the end-to-end construct that orchestrates the flow of data into, and output from, a machine learning model (or set of multiple models). It includes raw data input, features, outputs, machine learning model and model parameters, and prediction outputs.

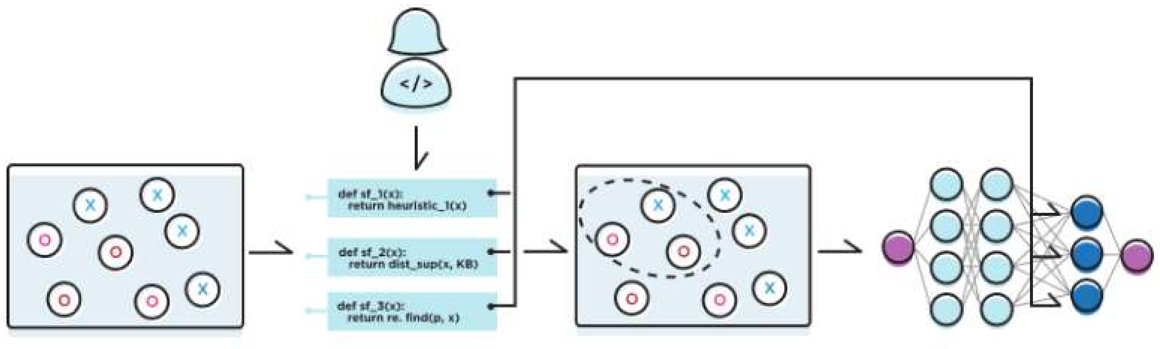
Specifying slices for *fine-grained* performance monitoring

Slice-based learning is a programming model for improving performance on application-critical data subsets, or slices.

In machine learning applications, some model predictions are more important than others — a subset of our data might correspond to safety-critical settings in an autonomous driving task (e.g. detecting cyclists) or a critical but low-frequency healthcare demographic (e.g. younger patients with certain cancers). However, learning objectives are often configured to optimize for overall quality metrics, which tend to be *coarse-grained*. In addition to overall performance, we'd like to monitor and improve *fine-grained* model performance on application-critical subsets, which we call *slices*.

Technical challenges of slice-based learning

- **Coping with noise:** SFs are specified as weak supervision; the model will need to be robust to noisy SF outputs.
- **Scalability:** As we add more slices to the model, we must maintain a parameter-efficient representation (as opposed to making multiple copies of large models).
- **Stable improvement of the model:** As the number of slices scale, we don't want to worsen performance on existing slices.



```
def sf_1(x):
  return heuristic_1(x)
```

```
def sf_2(x):
  return dist_sup(x, KB)
```

```
def sf_3(x):
  return re. find(p, x)
```

Slice-based learning pipeline: A developer specifies model subsets via slicing functions, and those subsets are used to add targeted extra capacity to our backbone architecture.

Result- The result has been verified under experimental error conditions.

Experiment 06

Aim- Create multiple models and ensemble them using VotingClassifier to improve the classifier's accuracy.

Platform- Google Colaboratory

Theory-

Ensemble model combine the decisions from multiple models to improve the overall performance.

Simple Ensemble Methods

Mode: In statistical terminology, "mode" is the number or value that most often appears in a dataset of numbers or values. In this ensemble technique, machine learning professionals use a number of models for making predictions about each data point. The predictions made by different models are taken as separate votes. Subsequently, the prediction made by most models is treated as the ultimate prediction.

The Mean/Average: In the mean/average ensemble technique, data analysts take the average predictions made by all models into account when making the ultimate prediction.

The Weighted Average: In the weighted average ensemble method, data scientists assign different weights to all the models in order to make a prediction, where the assigned weight defines the relevance of each model. As an example, let's assume that out of 100 people who gave feedback for your travel app, 70 are professional app developers, while the other 30 have no experience in app development. In this scenario, the weighted average ensemble technique will give more weight to the feedback of app developers compared to others.

Advanced Ensemble Methods

Bagging (Bootstrap Aggregating): The primary goal of "bagging" or "bootstrap aggregating" ensemble method is to minimize variance errors in decision trees. The objective here is to randomly create samples of training datasets with replacement (subsets of the training data). The subsets are then used for training decision trees or models. Consequently, there is a combination of multiple models, which reduces variance, as the average prediction generated from different models is much more reliable and robust than a single model or a decision tree.

Boosting: An iterative ensemble technique, "boosting," adjusts an observation's weight based on its last classification. In case observation is incorrectly classified, "boosting" increases the observation's weight, and vice versa. Boosting algorithms reduce bias errors and produce superior predictive models.

In the boosting ensemble method, data scientists train the first boosting algorithm on an entire dataset and then build subsequent algorithms by fitting residuals from the first boosting algorithm, thereby giving more weight to observations that the previous model predicted inaccurately.

Voting Classifier

A Voting Classifier is a machine learning model that trains on an ensemble of numerous models and predicts an output (class) based on their highest probability of chosen class as the output. It simply aggregates the findings of each classifier passed into Voting Classifier and predicts the output class based on the highest majority of voting. The idea is instead of creating separate dedicated models and finding the accuracy for each them, create a single model which trains by these models and predicts output based on their combined majority of voting for each output class.

Voting Classifier supports two types of votings.

1. **Hard Voting:** In hard voting, the predicted output class is a class with the highest majority of vote's i.e. the class which had the highest probability of being predicted by each of the classifiers. Suppose three classifiers predicted the *output class (A, A, B)*, so here the majority predicted *A* as output. Hence *A* will be the final prediction.
2. **Soft Voting:** In soft voting, the output class is the prediction based on the average of probability given to that class. Suppose given some input to three models, the prediction probability for class *A = (0.30, 0.47, 0.53)* and *B = (0.20, 0.32, 0.40)*. So the average for class *A is 0.4333* and *B is 0.3067*, the winner is clearly class *A* because it had the highest probability averaged by each classifier.

Result- The result has been verified under experimental error conditions.

Experiment 07

Aim- With a tree-based model having nominal (unordered) features implement OrdinalEncoder and demonstrate that OrdinalEncoder is faster than the OneHotEncoder and accuracy is also the same.

Platform- Google Colaboratory

Theory-

The word “Data” arises from the Latin word “Datum,” which means “something given.” **There are two classes of data: Qualitative and Quantitative data**, which are further classified into **four types: nominal, ordinal, discrete, and Continuous.**

Qualitative or Categorical Data

Qualitative or Categorical Data is data that can’t be measured or counted in the form of numbers. These types of data are sorted by category, not by number. That’s why it is also known as Categorical Data. These data consist of audio, images, symbols, or text. The gender of a person, i.e., male, female, or others, is qualitative data.

Qualitative data tells about the perception of people. This data helps market researchers understand the customers’ tastes and then design their ideas and strategies accordingly.

The other examples of qualitative data are:

- What language do you speak
- Favourite holiday destination
- Opinion on something (agree, disagree, or neutral)
- Colours

The Qualitative data are further classified into two parts:

Nominal Data

Nominal Data is used to label variables without any order or quantitative value. The colour of hair can be considered nominal data, as one colour can’t be compared with another colour.

The name “nominal” comes from the Latin name “nomen,” which means “name.” With the help of nominal data, we can’t do any numerical tasks or can’t give any order to sort the data. These data don’t have any meaningful order; their values are distributed to distinct categories.

Examples of Nominal Data:

- Colour of hair (Blonde, red, Brown, Black, etc.)
- Marital status (Single, Widowed, Married)
- Nationality (Indian, German, American)
- Gender (Male, Female, Others)
- Eye Color (Black, Brown, etc.)

Ordinal Data

Ordinal data have natural ordering where a number is present in some kind of order by their position on the scale. These data are used for observation like customer satisfaction, happiness, etc., but we can't do any arithmetical tasks on them. The ordinal data is qualitative data for which their values have some kind of relative position. These kinds of data can be considered as "in-between" the qualitative data and quantitative data. The ordinal data only shows the sequences and cannot use for statistical analysis. Compared to the nominal data, ordinal data have some kind of order that is not present in nominal data.

Examples of Ordinal Data:

- When companies ask for feedback, experience, or satisfaction on a scale of 1 to 10
- Letter grades in the exam (A, B, C, D, etc.)
- Ranking of peoples in a competition (First, Second, Third, etc.)
- Economic Status (High, Medium, and Low)
- Education Level (Higher, Secondary, Primary)

Quantitative Data

Quantitative data can be expressed in numerical values, which makes it countable and includes statistical data analysis. These kinds of data are also known as numerical data. It answers the questions like, "how much," "how many," and "how often." For example, the price of a phone, the computer's ram, the height or weight of a person, etc., falls under the quantitative data. Quantitative data can be used for statistical manipulation and these data can be represented on a wide variety of graphs and charts such as bar graphs, histograms, scatter plots, boxplot, pie charts, line graphs, etc.

Examples of Quantitative Data:

- Height or weight of a person or object
- Room Temperature
- Scores and Marks (Ex: 59, 80, 60, etc.)
- Time

The Quantitative data are further classified into two parts:

Discrete Data

The term discrete means distinct or separate. The discrete data contain the values that fall under integers or whole numbers. The total number of students in a class is an example of discrete data. These data can’t be broken into decimal or fraction values.

The discrete data are countable and have finite values; their subdivision is not possible. These data are represented mainly by a bar graph, number line, or frequency table.

Examples of Discrete Data:

- Total numbers of students present in a class
- Cost of a cell phone
- Numbers of employees in a company
- The total number of players who participated in a competition
- Days in a week

Continuous Data

Continuous data are in the form of fractional numbers. It can be the version of an android phone, the height of a person, the length of an object, etc. Continuous data represents information that can be divided into smaller levels. The continuous variable can take any value within a range.

The key difference between discrete and continuous data is that discrete data contains the integer or whole number. Still, continuous data stores the fractional numbers to record different types of data such as temperature, height, width, time, speed, etc.

Examples of Continuous Data:

- Height of a person
- Speed of a vehicle
- "Time-taken" to finish the work
- Wi-Fi Frequency
- Market share price

Many machine learning models, such as regression or SVM, are algebraic. This means that their input must be numerical. To use these models, categories must be transformed into numbers first, before you can apply the learning algorithm on them.

One Hot Encoder

What one hot encoding does is, it takes a column which has categorical data, which has been label encoded, and then splits the column into multiple columns. The numbers are replaced by 1s and 0s, depending on which column has what value.

Label Encoding

Another approach is to encode categorical values with a technique called "label encoding", which allows you to convert each value in a column to a number. Numerical labels are always between 0 and n_categories-1.

Binary Encoding

This technique is not as intuitive as the previous ones. In this technique, first the categories are encoded as ordinal, then those integers are converted into binary code, then the digits from that binary string are split into separate columns. This encodes the data in fewer dimensions than one-hot.

Result- The result has been verified under experimental error conditions.

Experiment 08

Aim- Implement feature encoders to encode categorical features:

- OneHotEncoder for unordered (nominal) data
- OrdinalEncoder for ordered (ordinal) data

Platform- Google Colaboratory

Theory-

Machine learning models can only work with numerical values. For this reason, it is necessary to transform the categorical values of the relevant features into numerical ones. This process is called *feature encoding*.

Data frame analytics automatically performs feature encoding. The input data is pre-processed with the following encoding techniques:

- One-hot encoding: Assigns vectors to each category. The vector represent whether the corresponding feature is present (1) or not (0).
- Target-mean encoding: Replaces categorical values with the mean value of the target variable.
- Frequency encoding: Takes into account how many times a given categorical value is present in relation with a feature.

When the model makes predictions on new data, the data needs to be processed in the same way it was trained. The category_encoders is a very useful library for encoding categorical columns. It supports one-hot, binary and label encoding techniques. You can install category_encoders via pip install category_encoders on cmd or just download and extract the .tar.gz file from the site.

You have to first import the category_encoders library after installing it. Invoke the BinaryEncoder function by specifying the columns you want to encode and then call the .fit_transform () method on it with the Data Frame as the argument.

Result- The result has been verified under experimental error conditions.

Experiment 09

Aim- Implement feature selection to a Pipeline:

1. Use SelectPercentile to keep the highest scoring features
2. Add feature selection after preprocessing but before model building

Platform- Google Colaboratory

Theory-

Feature Selection is the process of selecting out the most significant features from a given dataset. In many of the cases, Feature Selection can enhance the performance of a machine learning model as well.

Feature selection is also known as **Variable selection** or **Attribute selection**.

Essentially, it is the process of selecting the most important/relevant features of a dataset. The importance of feature selection can best be recognized when you are dealing with a dataset that contains a vast number of features. This type of dataset is often referred to as a *high dimensional* dataset. Now, with this high dimensionality, comes a lot of problems such as - this high dimensionality will significantly increase the training time of your machine learning model, it can make your model very complicated which in turn may lead to Over fitting.

Often in a high dimensional feature set, there remain several features which are redundant meaning these features are nothing but extensions of the other essential features. These redundant features do not effectively contribute to the model training as well. So, clearly, there is a need to extract the most important and the most relevant features for a dataset in order to get the most effective predictive modeling performance.

Sometimes, feature selection is mistaken with dimensionality reduction. But they are different. Feature selection is different from dimensionality reduction. Both methods tend to reduce the number of attributes in the dataset, but a dimensionality reduction method does so by creating new combinations of attributes (sometimes known as feature transformation), whereas feature selection methods include and exclude attributes present in the data without changing them.

Result- The result has been verified under experimental error conditions.