**Predicting Personal Loan Approval**

Using Machine Learning

THIRUVALLUVAR GOVERNMENT ARTS

COLLEGE RASIPURAM



TEAM LEADER: ASHWITHA E (20UCS2290)

TEAM MEMBER: KAVINA P (20UCS2292)

TEAM MEMBER: NITHYA A (20UCS2293)

TEAM MEMBER: SOWMIYA D (20UCS2295)

**PREDICTING PERSONAL LOAN APPROVAL USING MACHINE LEARNG**

**Milestone 1: Define Problem/Problem Understanding**

**Activity1: Specify the business problem**

**Activity2: Business Requirements**

**Activity3: Literature Survey (Student will write)**

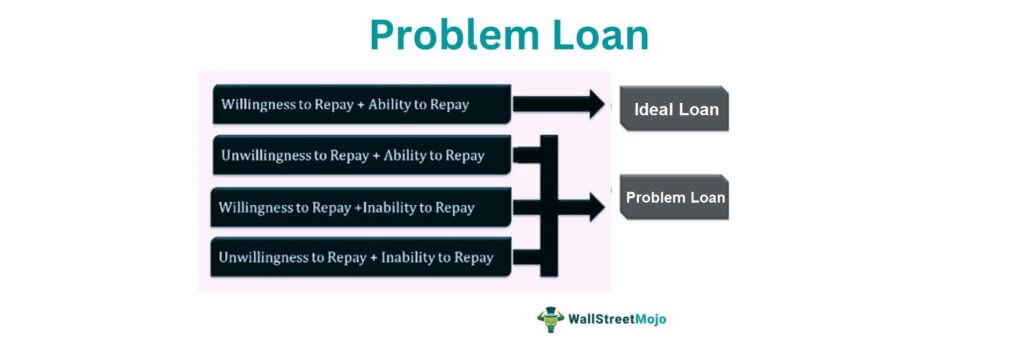
**Activity4: Social or Business Impact.**

**Activity 1: Business Problem**

# Introduction Loan Prediction Problem

Welcome to this article on Loan Prediction Problem. Below is a brief introduction to this topic to get you acquainted with what you will be learning.

A problem loan is a scenario where borrowers fail to repay monthly loan installments. The bank labels these loans as nonperforming assets (NPA). It can occur with either a commercial loan or a consumer loan.



# Problem Statement

Understanding the problem statement is the first and foremost step. This would help you give an intuition of what you will face ahead of time. Let us see the problem statement.

Dream Housing Finance company deals in all home loans. They have a presence across all urban, semi-urban and rural areas. Customers first apply for a home loan after that company validates the customer’s eligibility for a loan. The company wants to automate the loan eligibility process (real-time) based on customer detail provided while filling out the online application form. These details are Gender, Marital Status, Education, Number of Dependents, Income, Loan Amount, Credit History, and others. To automate this process, they have given a problem to identify the customer segments, that are eligible for loan amounts so that they can specifically target these customers.



It is a classification problem where we have to predict whether a loan would be approved or not. In these kinds of problems, we have to predict discrete values based on a given set of independent variables (s). Classification can be of two types:

* **Binary Classification:** In this, we have to predict either of the two given classes. For example: classifying the “gender” as male or female, predicting the “result” as to win or loss, etc.
* **MultiClass Classification**: Here we have to classify the data into three or more classes. For example: classifying a “movie’s genre” as comedy, action, or romantic, classifying “fruits” like oranges, apples, pears, etc.

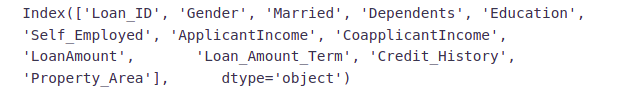
Loan prediction is a very common real-life problem that each retail bank faces at least once in its lifetime. If done correctly, it can save a lot of man-hours at the end of a retail bank.

Although this course is specifically built to give you a walkthrough of the Loan Prediction problem, you can always refer to the content to get a comprehensive overview to solve a classification problem.

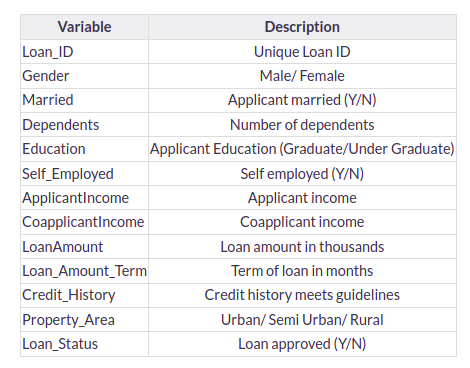
# Understanding the Problem

In this section, we will look at the train and test datasets. Firstly, we will check the features present in our data, and then we will look at their data types.

test.columns



We have similar features in the test dataset as the training dataset except for the Loan\_Status. We will predict the Loan\_Status using the model built using the train data. Given below is the description for each variable.



# Print data types for each variable

* object: Object format means variables are categorical. Categorical variables in our dataset are: Loan\_ID, Gender, Married, Dependents, Education, Self\_Employed, Property\_Area, Loan\_Status
* int64: It represents the integer variables. Applicant Income is of this format.
* float64: It represents the variable that has some decimal values involved. They are also numerical variables. Numerical variables in our dataset are: Co applicant Income, Loan Amount, Loan\_Amount\_Term, and Credit\_History

Let’s look at the shape of the dataset.

Train .shape, test .shape

https://editor.analyticsvidhya.com/uploads/19813Screenshot%20from%202022-05-11%2022-04-39.png

We have 614 rows and 13 columns in the training dataset and 367 rows and 12 columns in the test dataset

**Activity 2 : Business Requirements**

# INTRODUCTION:

**ACTIVITY**

A business requirements document is a report detailing everything a new project requires for success. This document outlines project objectives, what’s expected throughout the project lifecycle, and what’s required to accomplish the project.

# CREDIT SCORE AND HISTOR:

An applicant’s credit score is one of the most important factors a lender considers when evaluating a lone application. Credit scores range from 300 to 850 and are based on factors like payment history amount of outstanding debt and length of credit history. Many lenders require applicants to have a minimum score of around 600 to qualify, but some lenders will lend to applicants without any credit history at all.

# TO- DEBIT-INCOME RATIO:

Debt –to – income ratio (DDTI) is expressed as a percentage and represents the portion of a borrower’s gross monthly income that goes toward her monthly debt service. Lenders use DTI to predict a prospective borrower’s ability to make payments on new and current debt. For that reason, a DTI less than 36% is ideal, though some lenders will approve a highly qualified applicant a ratio up to 50%.

# COLLATERAL:

If you’re applying for a secured personal loan, your lender will require you to pledge valuable assets\_\_ or collateral. In the case of loans for homes or vehicles, the collateral is typically related to the underlying purpose of the loan. However, secured personal loans can also be collateralized by other valuable assets, including cash accounts, investment accounts, real estate and collectibles like coins or precious metals.

# ORIGINATION FEE:

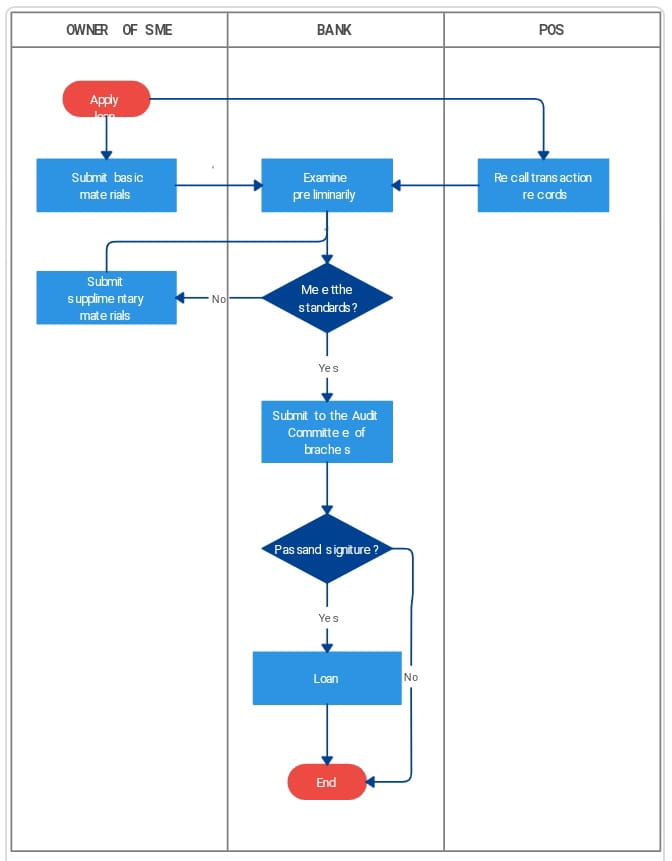
Though not part of the qualification process, many lenders require borrowers to pay personal loan origination fees to cover the costs of processing applications, running credit checks and closing. These fees usually range between 1% and 8% of the total loan amount, depending of factors like the applicant’s credit score and loan amount.

# TYPICAL PERSONAL LONE DOCUMENTS:

When it’s time to formally apply for a personal loan, your lender will request a number of documents to confirm everything from your identity to your residence and employment. Here are the most common documents lenders require as part of the personal loan application process.

# LOAN APPLICATION:

A loan application is a formal document that lenders require prospective borrowers to complete and submit to begin the lending process each lender has its own application, so the specific requirements may vary. In general, though, you’ll need to provide basic personal information, how much you want to borrow and the purpose of the loan.



# ROOF OF IDENTITY:

Most lenders require applicants to provide at least two forms of government- issued identification to prove they are at least 18 years old and a United States citizen.

* Driver’s license
* Other state-issued ID
* Passport
* Certificate of citizenship
* Birth certificate
* Military ID

# EMPLOYER AND INCOME VERIFICATION:

A lender wants to see that you have the ability to pay back your current debts as well as the new loan. To do this, lenders typically require prospective borrowers to demonstrate their employment history and current earnings part of the application process. Common forms of income verification for traditional employment include:

* Paystubs
* Returns
* W-2s and 1099s
* Bank statements
* Employer contact information

# PROOF OF ADDRESS:

In addition to confirming your employment, most lenders want to know that you have a stable living situation. This may involve providing proof of your address, including a recent utility bill, a copy of your lease or other rental agreement, auto insurance that lists your address.

# CONCIUSION:

Your conclusion should; reiterate the opportunity, highlight the key strengths of your plan and remind the reader why your business is a position to successfully execute the plan. If you are looking to raise funding with your plan, you shout detail finance required.

**ACTIVITY 3: LITERATURE SURVEY**

# INTRODUCTION

As the data are increasing daily due to digitization in the banking sector, people want to apply for loans through the internet. Artificial intelligence (AI), as a typical method for information investigation, has gotten more consideration increasingly. Individuals of various businesses are utilizing **AI** calculations to take care of the issues dependent on their industry information. Banks are facing a significant problem in the approval of the loan. Most banks can profit from the loan, but it is risky to choose deserving customers from the number of applications. Loan distribution is the primary business of almost every bank. This project aims to provide a loan [1, 8] to a deserving applicant out of all applicants. The bank authorities complete all other customer’s other formalities on time, which positively impacts the customers.

What Do Banks Broadly Check?

* **CIBIL Score and Report:** It is one of the most important factor that affects your loan approval. A good credit score and report is a positive indicator of your credit health.
* **Employment Status:** Apart from a good credit history, lenders also check for your steady income and employment status.
* **Account details:** Suit field or written off cases are carefully examined by lenders.
* **Payment History:** Lenders check for any default on payments or amount overdue cases, which might project a negative overview of your overall report.
* **EMI to Income Ratio:** Banks also consider the proportion of your existing loans when compared to your salary at the time of loan application. Your chances of loan approval get reduced if your total EMI’s exceed your monthly salary by 50%.

# LITERATURE SURVEY

A prediction is a statement about what someone thinks will happen in the future. People make predictions all the time. Some are very serious and are based on scientific calculations, but many are just guesses. Prediction helps us in many things to guess what will happen after some time or after a year ten years.

* Manjeet et al (2018) [24] there are seven types of variables that may influence consumer loan default; consumer‘s annual income, debit –income ratio, occupation, home ownership, work duration and whether or not consumer possesses a saving /checking account.
* In 2019, Vimala and sharmili [1] proposed a loan prediction model using and support Vector Machines (SVM) methods. Nave Bayes, an independent speculation approach, encompasses probability theory regarding the data classification. On the other hand, SVM uses statistical learning model for classification of prediction.
* In 2019, Jency, Sumathi and Shiva Sri [2] proposed an Exploratory Data Analysis (EDA) regarding the loan prediction procedure based on the client’s nature and their requirements. Dataset from UCI repository with 21 attributes was adopted to evaluate the proposed method. Finally, the outcome of the present work was to infer the constraints on the customer who are applying for the loan followed by the prediction regarding the repayment.
* In 2017, Goyal and Kaur [4] presented a loan prediction model using several Machine Learning (ML) algorithms. The dataset with features , namely ,gender, marital status, education, number of dependents, employment status, income, co applicant’s income, loan amount, loan tenure, credit history, existing loan status, and property area, are used for determining the loan eligibility regarding the loan sanctioning process.

### CONCLUSION

This application is working properly and meeting to all banker requirements. This component can be easily plugged in many other system. It works correctly and fulfills all requirement of bankers and can be connected to many other system. There were multiple malfunctions in the computers, content errors and fixing of weight in computerized prediction system.

**ACTIVITY 4: SOCIAL OR BUSINESS IMPACT**

From a social standpoint, using machine learning for personal loan approval can lead to greater efficiency and fairness in the lending process. By automating the loan approval process, lenders can reduce the time it takes to approve loans and ensure that all loan applications are evaluated consistently, without bias. This can be particularly beneficial for individuals who may have faced discrimination or bias in the past, such as those from marginalized communities.

On the other hand, there is also a potential for negative social impact if the machine learning algorithms used for loan approval are biased or discriminatory. It is important for lenders to ensure that their algorithms are unbiased and that they are not perpetuating any discriminatory practices or biases.

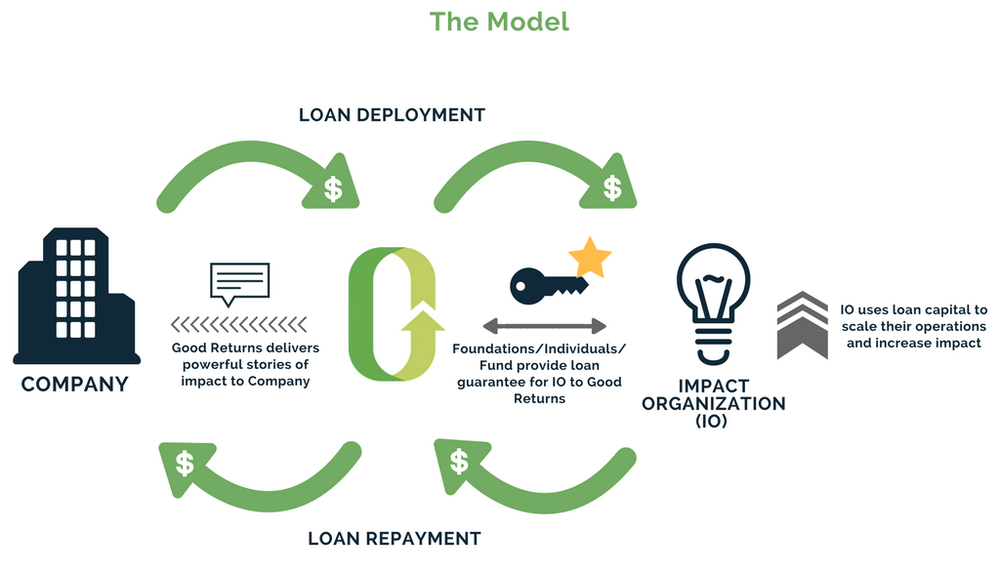
From a business standpoint, using machine learning for personal loan approval can lead to increased profitability and reduced risk for lenders. By accurately predicting the likelihood of loan repayment, lenders can more effectively manage their loan portfolios and reduce their risk of default.Additionally, the use of machine learning can help lenders identify new customers and expand their customer base.

However, there is also a potential for negative business impact if the machine learning algorithms used for loan approval are not accurate or effective. Inaccurate predictions can lead to higher default rates and increased risk for lenders, which can ultimately impact their profitability.

In summary, the use of machine learning for personal loan approval can have both positive and negative social and business impacts. It is important for lenders to carefully evaluate the potential impact of their machine learning algorithms and take steps to ensure that they are fair, accurate, and effective

Social or Business Impact for Predicting Personal Loan Approval Using Machine Learning

The use of machine learning to predict personal loan approval can have significant social and business impacts. Here are some possible outcomes:

Social Impact

**Fairness:** Machine learning algorithms can eliminate human biases and make the loan approval process more fair and transparent, promoting equality and diversity in the financial sector.

# Accessibility: Using machine learning to automate loan approval processes can make it easier and faster for people to obtain loans, especially those who are underbanked or have limited access to financial services.

# Trust: Predicting loan approvals using machine learning can increase trust in financial institutions by providing accurate and unbiased decisions, which can help build long-term customer relationships.

# Business Impact

# 

# Efficiency: Machine learning can automate loan approval processes, which can save time and resources for financial institutions, making their operations more efficient and cost-effective.

# Risk Management: Predictive models can help financial institutions assess and manage the risk of loan defaults, minimizing losses and improving profitability.

# Customer Experience: Automated loan approval processes can improve the customer experience by providing fast and reliable loan decisions, which can increase customer satisfaction and loyalty.

# Overall, the use of machine learning for predicting personal loan approvals can have positive impacts on both society and businesses, by promoting fairness, accessibility, trust, efficiency, risk management, and customer experience. However, it is important to ensure that the algorithms used are transparent, unbiased, and comply with ethical and legal standards to prevent unintended consequences or harm.

**Milestone 2: Data Collection & Preparation**

**Activity1: Collect the Dataset**

**Activity 2: Importing the libraries**

**Activity 3: Read the dataset**

**Activity 4: Data Preparation**

**Activity 5: Handling Missing Values**

**Activity 6: Handling Categorical Values**

**Activity 7: Handling Imbalance Data**

**Activity 1: Collect the Dataset**

LOANS are the major requirement of the modern world. By this only, Banks get a major part of the total profit. It is beneficial for students to manage their education and living expenses, and for people to buy any kind of luxury like houses, cars, etc.

But when it comes to deciding whether the applicant’s profile is relevant to be granted with loan or not. Banks have to look after many aspects.

So, here we will be using Machine Learning with [Python](https://www.geeksforgeeks.org/python-programming-language/) to ease their work and predict whether the candidate’s profile is relevant or not using key features like Marital Status, Education, Applicant Income, Credit History, etc.

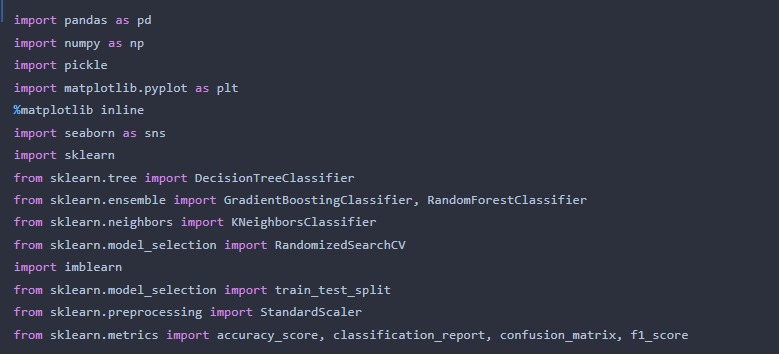
|  |  |  |
| --- | --- | --- |
| **1** | Loan | A unique id |
| **2** | Gender | Gender of the applicant Male/female |
| **3** | Married | Marital Status of the applicant, values will be Yes/ No |
| **4** | Dependents | It tells whether the applicant has any dependents or not. |
| **5** | Education | It will tell us whether the applicant is Graduated or not. |
| **6** | Self employed | This defines that the applicant is self-employed i.e. Yes/ No |
| **7** | ApplicantIncome | Applicant income |
| **8** | Co applicant Income | Co-applicant income |
| **9** | LoanAmount | Loan amount (in thousands) |
| **10** | Loan Amount term | Terms of loan (in months) |
| **11** | Credit History | Credit history of individual’s repayment of their debts |
| **12** | Property Area | Area of property i.e. Rural/Urban/Semi-urban |
| **13** | Loan Status | Status of Loan Approved or not i.e. Y- Yes, N-No |

**Activity 2:**

**Importing Libraries and Dataset**

Firstly we have to import libraries:

* [Pandas](https://www.geeksforgeeks.org/python-pandas-dataframe/) – To load the Data frame
* Matplotlib – To visualize the data features i.e. bar plot.
* [Seaborn](https://www.geeksforgeeks.org/introduction-to-seaborn-python/) – To see the correlation between features using heat map



**Activity3:Read the Data****set**

This kernel I want try to make a loan approval prediction based on [Loan Prediction](https://www.kaggle.com/ninzaami/loan-predication/home) dataset. Previously, I was made the same work for my university course task but with different dataset. You can see my previous work on [this link](https://github.com/hafidhfikri/Bankruptcy-Prediction-Model) (in Bahasa). This work will compare the result of loan approval prediction classification algorithm with different dataset from my previous work. In previous work I get the Gradient Boosting Classifier as the best classifier for loan approval prediction. beside that, I want to make some improvement in this kernel inspired by the work of  Balighs.

We have 12 independent variables and 1 target variable, i.e. Loan\_Status in the training dataset.

test.columnsIndex(['Loan\_ID', 'Gender', 'Married', 'Dependents', 'Education',  
 'Self\_Employed', 'ApplicantIncome', 'Co applicant Income', 'LoanAmount',  
 'Loan\_Amount\_Term', 'Credit\_History', 'Property\_Area'],  
 dtype='object')

We have similar features in the test dataset as the training dataset except for the Loan\_Status. We will predict the Loan\_Status using the model built using the train data.

Train.dtypesLoan\_ID object  
Gender object  
Married object  
Dependents object  
Education object  
Self\_Employed object  
ApplicantIncome int64  
CoapplicantIncome float64  
LoanAmount float64  
Loan\_Amount\_Term float64  
Credit\_History float64  
Property\_Area object  
Loan\_Status object  
dtype: object

**We can see there are three formats of data types:**

**object:** Object format means variables are categorical. Categorical variables in our dataset are Loan\_ID, Gender, Married, Dependents, Education, Self\_Employed, Property\_Area, Loan\_Status.

**Categorical features**: These features have categories (Gender, Married, Self\_Employed, CreditHistory, Loan\_Status)

* **Ordinal features**: Variables in categorical features having some order involved (Dependents, Education, Property\_Area)
* **Numerical features**: These features have numerical values (ApplicantIncome, Co-applicantIncome, Loan Amount, Loan \_Amount\_ Term

|  | Loan\_ID | Gender | Married | Dependents | Education | Self employed | ApplicantIncome | CoapplicantIncome | LoanAmount | Loan Amount Term | Credit\_History | Property\_Area | Loan\_Status |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | LP001002 | Male | No | 0 | Graduate | No | 5849 | 0.0 | NaN | 360.0 | 1.0 | Urban | Y |
| 1 | LP001003 | Male | Yes | 1 | Graduate | No | 4583 | 1508.0 | 128.0 | 360.0 | 1.0 | Rural | N |
| 2 | LP001005 | Male | Yes | 0 | Graduate | Yes | 3000 | 0.0 | 66.0 | 360.0 | 1.0 | Urban | Y |
| 3 | LP001006 | Male | Yes | 0 | Not Graduate | No | 2583 | 2358.0 | 120.0 | 360.0 | 1.0 | Urban | Y |
| 4 | LP001008 | Male | No | 0 | Graduate | No | 6000 | 0.0 | 141.0 | 360.0 | 1.0 | Urban | Y |

## Activity 4: data preparation

## Data preparation is the process of preparing raw data so that it is suitable for further processing and analysis. Key steps include collecting, cleaning, and labelling raw data into a form suitable for machine learning (ML) algorithms and then exploring and visualizing the data. Data preparation can take up to 80% of the time spent on an ML project. Using specialized data preparation tools is important to optimize this process.

Data preparation follows a series of steps that starts with collecting the right data, followed by cleaning, labelling, and then validation and visualization.

### Collect data

### Collecting data is the process of assembling all the data you need for ML. Data collection can be tedious because data resides in many data sources, including on laptops, in data warehouses, in the cloud, inside applications, and on devices. Finding ways to connect to different data sources can be challenging.

### Clean data

Cleaning data corrects errors and fills in missing data as a step to ensure data quality. After you have clean data, you will need to transform it into a consistent, readable format.

### Label data

Data labelling is the process of identifying raw data (images, text files, videos, and so on) and adding one or more meaningful and informative labels to provide context so an ML model can learn from it. For example, labels might indicate if a photo contains a bird or car, which words were mentioned in an audio recording, or if an X-ray discovered an irregularity.

### Validate and visualize

After data is cleaned and labelled, ML teams often explore the data to make sure it is correct and ready for ML. Visualizations like histograms, scatter plots, box and whisker plots, line plots, and bar charts are all useful tools to confirm data is correct. Additionally, visualizations also help data science terms complete exploratory data analysis.

**Activity 5: Handling Missing Values**

One way of handling missing values is the deletion of the rows or columns having null values. If any columns have more than half of the values as null then you can drop the entire column. In the same way, rows can also be dropped if having one or more columns value

## 1) A Simple Option: Drop Columns with Missing Values

If your data is in a DataFrame called original\_data, you can drop columns with missing values. One way to do that is

Data\_without\_missing\_values = original\_data.dropna(axis=1)

In many cases, you'll have both a training dataset and a test dataset. You will want to drop the same columns in both DataFrames. In that case, you would write

Colswith\_missing = [col for col in original\_data.columns

if original\_data[col].isnull().any()]

reduced original\_data = original\_data.drop(cols\_with\_missing, axis=1)

reduce \_test data = test data.drop(cols\_with\_missing, axis=1)

If those columns had useful information (in the places that were not missing), your model loses access to this information when the column is dropped. Also, if your test data has missing values in places where your training data did not, this will result in an error.

So, it's somewhat usually not the best solution. However, it can be useful when most values in a column are missing.

**2) A Better Option**: Imputation  you will learn three approaches to dealing with missing values. You will then learn to compare the effectiveness of these approaches on any given data set.\*

Python libraries represent missing numbers as **nan** which is short for "not a number". You can detect which cells have missing values, and then count how many there are in each column with the command:

Print (missing\_ val \_count by\_ column[missing Python libraries represent missing numbers as **nan** which is short for "not a number". You can detect which cells have missing values, and then count how many there are in each column with the command:

missing\_ val\_ count\_ by\_ column = (data .is null().sum())

print (missing \_val\_ count\_ by\_ column[missing \_val \_count\_ by\_ column > 0

Most libraries (including scikit-learn) will give you an error if you try to build a model using data with missing values. So you'll need to choose one of the strategies below

reduced\_original\_data = original numbers had useful information (in the places that were not missing), your model loses access to this information when the column is dropped. Also, if your test data has missing values in places where your training data

Imputation fills in the missing value with some number.

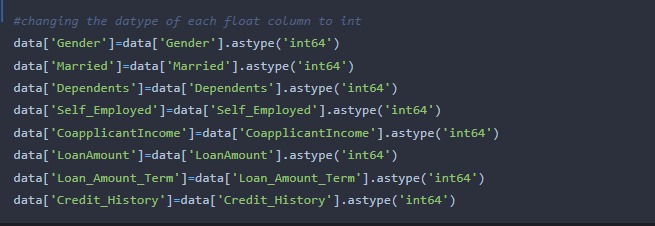
**3) An Extension To Imputation**

Imputation is the standard approach, and it usually works well. However, imputed values may by systematically above or below their actual values (which weren't collected in the dataset). Or rows with missing values may be unique in some other way. In that case, your model would make better predictions by considering which values were originally missing. Here's how it might look:

**Activity 6:Handling Categorical Values**

1. Using the categorical variable, evaluate the probability of the Target variable (where the output is True orfalse . 2) Calculate the probability of the Target variable having a 0 output. 3) Calculate the problem .

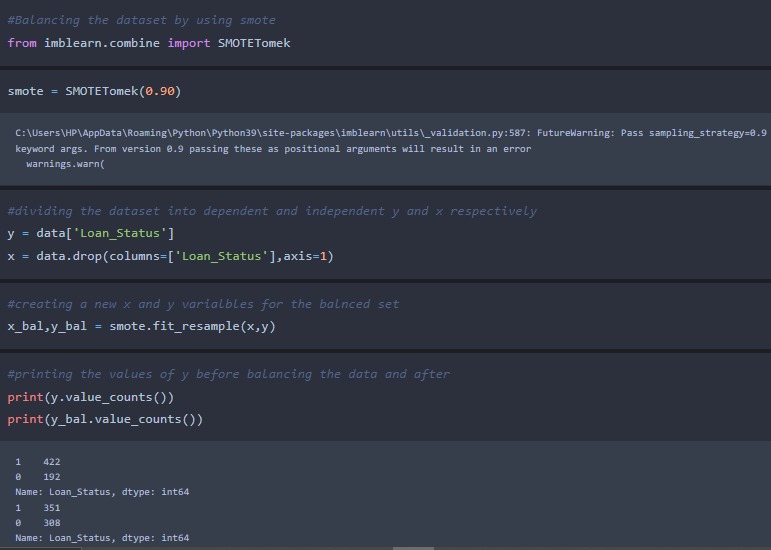
**Analyzing Categorical Features**

1. Value Counts. `value\_counts()` is a function in the pandas library that returns the frequency of each unique value in a categorical data column. ...
2. Group by. ...
3. Cross tab. ...
4. Pivot Table. ...
5. One-hot encoding.  
     
   We use this categorical data encoding technique when the categorical feature is ordinal. In this case, retaining the order is important. Hence encoding should reflect the sequence. In Label encoding, each label is converted into an integer value.

**Activity 7: Handling Imbalance Data**

Imbalanced data refers to those types of datasets where the target class has an uneven distribution of observations, i.e one class label has a very high number of observations and the other has a very low number of observations. We can better understand imbalanced dataset handling with an example.

Let’s assume that XYZ is a bank that issues a credit card to its customers. Now the bank is concerned that some fraudulent transactions are going on and when the bank checks their data they found that for each 2000 transaction there are only 30 Nos of fraud recorded. So, the number of fraud per 100 transactions is less than 2%, or we can say more than 98% transaction is “No Fraud” in nature. Here, the class “No Fraud” is called the **majority class,** and the much smaller in size “Fraud” class is called the **minority class**.

****

**Milestone 3:Exploratory Data Analysis**

**Activity1:Descriptive Statistical**

**Activity2:Visual Analysis**

**Activity3:Univariate Analysis**

**Activity4:Bivariate Analysis**

**Activity5:Multivariate Analysis**

**Activity6:Training The Model In Multiple Algorithms**

**Activity7:Decision Tree Model**

**Activity8:Random Forest Model**

**Activity9:KNN Model**

**Activity10:Testing The Model**

**Activity1:Descriptive Statistical**

By ADAM HAYES Updated March 21, 2023

Reviewed by THOMAS BROCK

Fact checked by MICHAEL LOGAN

Descriptive Statistics

Jessica Olah

**What Are Descriptive Statistics?**

Descriptive statistics are brief informational coefficients that summarize a given data set, which can be either a representation of the entire population or a sample of a population. Descriptive statistics are broken down into measures of central tendency and measures of variability (spread). Measures of central tendency include the mean, median, and mode, while measures of variability include standard deviation, variance, minimum and maximum variables, kurtosis, and skewness.

**KEY TAKEAWAYS :**

Descriptive statistics summarizes or describes the characteristics of a data set.Descriptive statistics consists of three basic categories of measures: measures of central tendency, measures of variability f(or spread), and frequency distribution.Measures of central tendency describe the center of the data set (mean, median, mode).Measures of variability describe the dispersion of the data set (variance, standard deviation).Measures of frequency distribution describe the occurrence of data within the data set (count).

**Activity 2:Visual Analysis**

**The essence of visual analysis**

Visual analysis increases your understanding of how visual material communicates and functions, whether it generates meaning, elicits emotion or creates a mood. Visual analysis can be applied to any visual material including art, design and architecture.

Visual analysis identifies and explores characteristics of example material and relationships within the context in which they were produced and encountered. The purpose of visual analysis is to make an argument based on visual evidence. There are three parts to writing a visual analysis:

identify, describe and analyse the visual material

situate the visual material in its context

interpret and respond to the content of the visual material

Identify, describe and analyse the visual material

Begin by stating the type of material (for example, a building, a photograph, etc.), who made it, its title/name, and the year it was created. If relevant, also state its media, materials, components, dimensions, and location.

Then, examine and describe formal elements such as colour, line, shape, texture, and tone.

Next, analyse the composition, for example, relationships between elements like balance, geometry, pattern, proportion, repetition, rhythm, scale, and symmetrical and asymmetrical balance.

You can also consider ways the artist conveys feelings through form and space, for example, dynamism, harmony and tension, illusion, light and shade, modeling, perspective, and positive and negative space.

**Note:** It is important to distinguish between description and analysis. Description merely describes or explains something, whereas analysis examines and evaluates relationships, and makes comparisons. You need to identify and describe the visual examples, but when you start discussing the composition and the feelings it conveys you’ll begin to analyse them.

**Form analysis**: Kazimir Malevich

Kazimir Malevich, Painterly Realism of a Football Player: Color Masses in the 4th Dimension, 1915, oil on canvas

Figure 1. Kazimir Malevich, Painterly Realism of a Football Player: Color Masses in the 4th Dimension, 1915, oil on canvas.

In the paintings of Russian modernist Kazimir Malevich, form is the content. Malevich’s Painterly Realism of a Football Player: Color Masses in the 4th Dimension, painted in 1915 (fig. 1), does not depict anything recognisable; it is not a picture of a football player. Form elements, especially colour, shape, proportion and depth, are expressive in their own right. The elements create an energetic tension between two ways of seeing the composition; sometimes the brightly-coloured geometric shapes look like they’re laying flat on a surface, but sometimes they look like they’re suspended in space.

Questions you could ask about Malevich’s painting to analyse it could include:

**How is the composition balanced?**

Does the painting look flat, or is there some feeling of deptend.

Do the shapes look randomly scattered or carefully arranged?

Check your understanding

**Countplot:-**

Acount plot can be thought of as a histogram across a categorical, instead of quantitative,

variable. The basic API and options are identical to those for barplot() , so you can compare

counts across nested variables.

From the graph we can infer that , gender and education is a categorical variables with 2

categories , from gender column we can infer that 0-category is having more weightage than

category-1,while education with 0,it means no education is a underclass when compared with

category -1, which means educated .

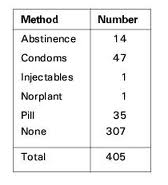
**Activity3:Univariate Analysis**

Univariate analysis explores each variable in a data set, separately. It looks at the range of values, as well as the central tendency of the values. It describes the pattern of response to the variable. It describes each variable on its own. Descriptive statistics describe and summarize data.

**What is a variable in Univariate Analysis?**

A variable in univariate analysis is just a condition or subset that your data falls into. You can think of it as a “category.” For example, the analysis might look at a variable of “age” or it might look at “height” or “weight”. However, it doesn’t look at more than one variable at a time otherwise it becomes bivariate analysis (or in the case of 3 or more variables it would be called multivariate analysis).

The following frequency distribution table shows one variable (left column) and the count in the right column.

[](https://www.statisticshowto.com/wp-content/uploads/2013/09/categorical-variable-frequency-distribution-table.jpg)

A frequency chart.

You could have more than one variable in the above chart. For example, you could add the variable “Location”or “Age” or something else, and make a separate column for location or age. In that case you would have bivariate data because you would then have two variables.

**Univariate Descriptive Statistics**

Some ways you can describe patterns found in univariate data include central tendency (mean, mode and median) and dispersion: range , variance, maximum, minimum, quartiles (including the interquartile range), and standard deviation.

You have several options for describing data with univariate data. Click on the link to find out more about each type of graph or chart:

Frequency Distribution Tables.

Bar Charts.

Histograms.

Frequency Polygons.

Pie Charts.

Check out our Statistics YouTube channel for hundreds of videos on elementary statistics, probability and more.

**Activity 4: Bivariate Analysis**

Bivariate analysis is one of the simplest forms of quantitative (statistical) analysis.[1] It involves the analysis of two variables (often denoted as X, Y), for the purpose of determining the empirical relationship between them.

Bivariate analysis can be helpful in testing simple hypotheses of association. Bivariate analysis can help determine to what extent it becomes easier to know and predict a value for one variable (possibly a dependent variable) if we know the value of the other variable (possibly the independent variable) (see also correlation and simple linear regression).

Bivariate analysis can be contrasted with univariate analysis in which only one variable is analysed.[1] Like univariate analysis, bivariate analysis can be descriptive or inferential. It is the analysis of the relationship between the two variables.[1] Bivariate analysis is a simple (two variable) special case of multivariate analysis (where multiple relations between multiple variables are examined simultaneously).

**When there is a dependent variable**

If the dependent variable—the one whose value is determined to some extent by the other, independent variable— is a categorical variable, such as the preferred brand of cereal, then probit or logit regression (or multinomial probit or multinomial logit) can be used. If both variables are ordinal, meaning they are ranked in a sequence as first, second, etc., then a rank correlation coefficient can be computed. If just the dependent variable is ordinal, ordered probit or ordered logit can be used. If the dependent variable is continuous—either interval level or ratio level, such as a temperature scale or an income scale—then simple regression can be used.

If both variables are time series, a particular type of causality known as Granger causality can be tested for, and vector autoregression can be performed to examine the intertemporal linkages between the variables.

**When there is not a dependent variable**

When neither variable can be regarded as dependent on the other, regression is not appropriate but some form of correlation analysis may be. [3]

Graphical methods

Graphs that are appropriate for bivariate analysis depend on the type of variable. For two continuous variables, a scatterplot is a common graph. When one variable is categorical and the other continuous, a box plot is common and when both are categorical a mosaic plot is common. These graphs are part of descriptive statistics.

See also

Canonical correlation

Coding (social sciences)

Descriptive statistics

External links

Discriminant correlation analysis (DCA)[4]

**Reference**

Earl R. Babbie, The Practice of Social Research, 12th edition, Wadsworth Publishing, 2009, ISBN 0-495-59841-0, pp. 436–440

Bivariate Analysis, Sociology Index>

Chatterjee, Samprit (2012). Regression analysis by example. Hoboken, New Jersey: Wiley. ISBN 978-0470905845.

M. Haghighat, M. Abdel-Mottaleb, & W. Alhalabi (2016). Discriminant Correlation Analysis: Real-Time Feature Level Fusion for Multimodal Biometric Recognition. IEEE Transactions on Information Forensics and Security, 11(9), 1984-1996.

Statistics

OutlineIndex

Descriptive statistics

Data collection

Statistical inference

CorrelationRegression analysis

Categorical / Multivariate / Time-series / Survival analysis

Applications

Categoryicon Mathematics portalCommons WikiProject

Categories: Multivariate statistics2 (number)

**Activity 5: Multivariate Analysis**

Multivariate analysis is defined as: The statistical study of data where multiple measurements are made on each experimental unit and where the relationships among multivariate measurements and their structure are important.

**Overview**

Data analysis is one of the most useful tools when one tries to understand the vast amount of information presented to them and synthesise evidence from it. There are usually multiple factors influencing a phenomenon.

Of these, some can be observed, documented and interpreted thoroughly while others cannot. For example, in order to estimate the burden of a disease in society there may be a lot of factors which can be readily recorded, and a whole lot of others which are unreliable and, therefore, require proper scrutiny. Factors like incidence, age distribution, sex distribution and financial loss owing to the disease can be accounted for more easily when compared to contact tracing, prevalence and institutional support for the same. Therefore, it is of paramount importance that the data which is collected and interpreted must be done thoroughly in order to avoid common pitfalls.

2 boxes side by side. Box 1 has a scatter plot with a nearly horizontal red line through it. At the bottom it states R squared = 0.06. The second box has the same scatter plot and then joined up red lines which look like a person holding a dog. The red text in this box says Rexthor, The Dog-Bearer. Below these boxes is the statement "I don't trust linear regressions when it's harder to guess the direction of the correlation from the scatter plot than to find new constellations on it".

**Why does it sound so important?**

Data collection and analysis is emphasised upon in academia because the very same findings determine the policy of a governing body and, therefore, the implications that follow it are the direct product of the information that is fed into the system.

**Introduction**

In this blog, we will discuss types of data analysis in general and multivariate analysis in particular. It aims to introduce the concept to investigators inclined towards this discipline by attempting to reduce the complexity around the subject.

Analysis of data based on the types of variables in consideration is broadly divided into three categories:

**Univariate analysis**: The simplest of all data analysis models, univariate analysis considers only one variable in calculation. Thus, although it is quite simple in application, it has limited use in analysing big data. E.g. incidence of a disease.

**Bivariate analysis**: As the name suggests, bivariate analysis takes two variables into consideration. It has a slightly expanded area of application but is nevertheless limited when it comes to large sets of data. E.g. incidence of a disease and the season of the year.

**Multivariate analysis**: Multivariate analysis takes a whole host of variables into consideration. This makes it a complicated as well as essential tool. The greatest virtue of such a model is that it considers as many factors into consideration as possible. This results in tremendous reduction of bias and gives a result closest to reality. For example, kindly refer to the factors discussed in the “overview” section of this article.

**Multivariate analysis is defined as:**

* The statistical study of data where multiple measurements are made on each experimental unit and where the relationships among multivariate measurements and their structure are important
* Multivariate statistical methods incorporate several techniques depending on the situation and the question in focus. Some of these methods are listed below:
* **Regression analysis**: Used to determine the relationship between a dependent variable and one or more independent variable.
* **Analysis of Variance (ANOVA):** Used to determine the relationship between collections of data by analyzing the difference in the means.
* **Interdependent analysis**: Used to determine the relationship between a set of variables among themselves.
* **Discriminant analysis**: Used to classify observations in two or more distinct set of categories.
* **Classification and cluster analysis**: Used to find similarity in a group of observations.
* **Principal component analysis**: Used to interpret data in its simplest form by introducing new uncorrelated variables.
* **Factor analysis**: Similar to principal component analysis, this too is used to crunch big data into small, interpretable forms.
* **Canonical correlation analysis**: Perhaps one of the most complex models among all of the above, canonical correlation attempts to interpret data by analysing relationships between cross-covariance matrices.
* ANOVA remains one of the most widely used statistical models in academia. Of the several types of ANOVA models, there is one subtype that is frequently used because of the factors involved in the studies. Traditionally, it has found its application in behavioural research, i.e. Psychology, Psychiatry and allied disciplines. This model is called the Multivariate Analysis of Variance (MANOVA). It is widely described as the multivariate analogue of ANOVA, used in interpreting univariate data.

4 boxes side by side. 1st box has a stick man sitting at a desk with a hill shaped object which has the words 'Students T Distribution' on it. They are wiggling it on top of a bit of paper he is saying "Hmm". The 2nd box the same scene exists, but he is now saying "....Nope". In the 3rd box he has lifted off the hill shaped object and walking away from the desk with it. In the final box, he is placing a new object onto the desk which is a hill shape, but with many more peaks and troughs on it with the words 'Teachers' T Distribution' on it.

Interpretation of results is probably the most difficult part in the technique. The relevant results are generally summarized in a table with an associated text. Appropriate information must be highlighted regarding:

Multivariate test statistics used

Degrees of freedom

Appropriate test statistics used

Calculated p-value (p < x)

Reliability and validity of the test are the most important determining factors in such techniques.

**Applications**

Multivariate analysis is used in several disciplines. One of its most distinguishing features is that it can be used in parametric as well as non-parametric tests.

**Quick question: What are parametric and non-parametric tests?**

Parametric tests: Tests which make certain assumptions regarding the distribution of data, i.e. within a fixed parameter.

Non-parametric tests: Tests which do not make assumptions with respect to distribution. On the contrary, the distribution of data is assumed to be free of distribution.

2 column table. First column is "Parametric tests". Under this is the following list: Based on Interval/Ratio Scale; Outliers absent; Uniformly distributed data; equal variance; sample size is usually large. The second column is titled "Non parametric tests". The list below this is as follows: Based on Nominal/Ordinal scale; Outliers present; Non uniform data; Unequal variance; Small sample size.

Uses of Multivariate analysis: Multivariate analyses are used principally for four reasons, i.e. to see patterns of data, to make clear comparisons, to discard unwanted information and to study multiple factors at once. Applications of multivariate analysis are found in almost all the disciplines which make up the bulk of policy-making, e.g. economics, healthcare, pharmaceutical industries, applied sciences, sociology, and so on. Multivariate analysis has particularly enjoyed a traditional stronghold in the field of behavioural sciences like psychology, psychiatry and allied fields because of the complex nature of the discipline.

**Conclusion**

Multivariate analysis is one of the most useful methods to determine

effective in minimizing bias if a structured study design is employed. However, the complexity of the technique makes it a less sought-out model for novice research enthusiasts. Therefore, although the process of designing the study and interpretation of results is a tedious one, the techniques stand out in finding the relationships in complex situations.

**Activity 6:Training The Model In Multiple Algorithms**

A Gentle Introduction to Multiple-Model Machine Learning

by Jason Brownlee on May 12, 2021 in Ensemble Learning

Tweet Tweet Share

Last Updated on October 22, 2021

An ensemble learning method involves combining the predictions from multiple contributing models.

Nevertheless, not all techniques that make use of multiple machine learning models are ensemble learning algorithms.

It is common to divide a prediction problem into subproblems. For example, some problems naturally subdivide into independent but related subproblems and a machine learning model can be prepared for each. It is less clear whether these represent examples of ensemble learning, although we might distinguish these methods from ensembles given the inability for a contributing ensemble member to produce a solution (however weakly) to the overall prediction problem.

In this tutorial, you will discover multiple-model techniques for machine learning and their relationship to ensemble learning.

**After completing this tutorial, you will know**:

Multiple-model machine learning refers to techniques that use multiple models in some way that closely resembles ensemble learning.

Use of multiple models for multi-class classification and multi-output regression differ from ensembles in that no contributing member can solve the problem.

Mixture of experts might be considered a true ensemble method, although hybrid machine learning models are probably not ensemble learning methods.

**Tutorial Overview**

**This tutorial is divided into five parts; they are:**

* Multiple-Model Techniques
* Multiple Models for Multi-Class Classification
* Multiple Models for Multi-Output Regression
* Multiple Expert Models
* Hybrids Constructed From Multiple Models

**Multiple-Model Techniques**

Ensemble learning is concerned with approaches that combine predictions from two or more models.

We can characterize a model as an ensemble learning technique if it has two properties, such as:

**Comprising two or more models**.

Predictions are combined.

We might also suggest that the goal of an ensemble model is to improve predictions over any contributing member. Although a lesser goal might be to improve the stability of the model, e.g. reduce the variance in the predictions or prediction errors.

Nevertheless, there are models and model architectures that contain elements of ensemble learning methods, but it is not clear as to whether they may be considered ensemble learning or not.

For example, we might define an ensemble learning technique as being composed of two or more models. The problem is that there may be techniques that have more than two models, yet do not combine their predictions. Alternatively, they may combine their predictions in unexpected ways**.**

**Acitivity 7: Decision Tree Model**

**Decision tree model**

* Article
* Talk
* Read
* Edit
* View history
* Tools

**From Wikipedia, the free encyclopedia**

In computational complexity the decision tree model is the model of computation in which an algorithm is considered to be basically a decision tree, i.e., a sequence of queries or tests that are done adaptively, so the outcome of previous tests can influence the tests performed next.

Typically, these tests have a small number of outcomes (such as a yes–no question) and can be performed quickly (say, with unit computational cost), so the worst-case time complexity of an algorithm in the decision tree model corresponds to the depth of the corresponding decision tree. This notion of computational complexity of a problem or an algorithm in the decision tree model is called its decision tree complexity or query complexity.

Decision trees models are instrumental in establishing lower bounds for complexity theory for certain classes of computational problems and algorithms. Several variants of decision tree models have been introduced, depending on the computational model and type of query algorithms are allowed to perform.

For example, a decision tree argument is used to show that a comparison sort of

n items must take

log

?

(

)

n\log(n) comparisons. For comparison sorts, a query is a comparison of two items

{\displaystyle a,\,b}, with two outcomes (assuming no items are equal): either

<

a<b or

>

a>b. Comparison sorts can be expressed as a decision tree in this model, since such sorting algorithms only perform these types of queries**.**

**Milestone 4: Model Building**

**Activity 1: Testing Model With Multiple Evaluation Metrics**

**Activity 2: Compare The Model**

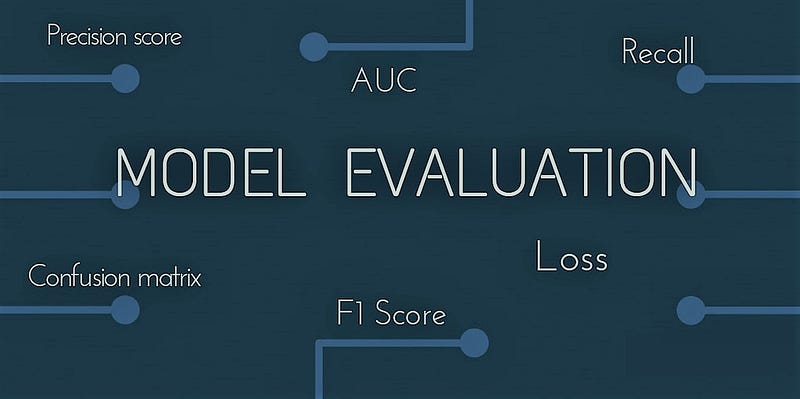
**Activity 3: Comparing Model Accuracy Before & After Applying Hyperparameter Tuning**

**Activity 1: Testing Model With Multiple Evaluation Metrics**

**What is model evaluation?**

**Model evaluation** is a process of assessing the model’s performance on a chosen evaluation setup. It is done by calculating quantitative performance metrics like F1 score or RMSE or assessing the results qualitatively by the subject matter experts. The machine learning **evaluation metrics you choose should reflect the business metrics** you want to optimize with the machine learning solution.

**Evaluation Metrics for Classification Problem**



**Abstract**

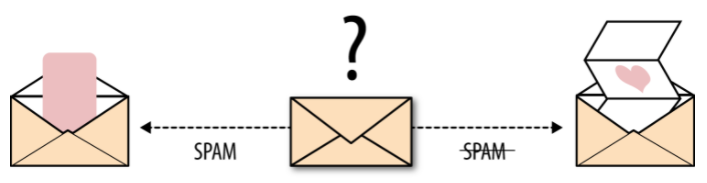
class 0.” The most important task in building any machine learning model is to evaluate its performance. So, the question arises that how would one measure the success of a machine learning model? How would we know that when to stop the training and evaluation and when to call it good?

With the help of this article, I’ll try to answer these questions. This article is organized as follows: In the first section, I’ll start with introducing what we mean by “ML Model Evaluation” and why it’s actually necessary to evaluate any machine learning model. In other subsequent section, we’ll discuss various evaluation metrics available to specific use cases in subsequent sections to better understand these metrics.

**Classification Metrics in Machine Learning**

Classification is about predicting the class labels given input data. In binary classification, there are only two possible output classes(i.e., Dichotomy). In multiclass classification, more than two possible classes can be present. I’ll focus only on binary classification.

A very common example of binary classification is spam detection, where the input data could include the email text and metadata (sender, sending time), and the output label is either *“spam” or “not spam.”* (*See Figure*) Sometimes, people use some other names also for the two classes: “positive” and “negative,” or “class 1” and “ list.



There are many ways for measuring classification performance. Accuracy, confusion matrix, log-loss, and AUC-ROC are some of the most popular metrics. Precision-recall is a widely used metrics for classification problems.

**Activity 2: Compare The Model**

### Application Areas

Automatic comparison is a must whenever you work with multiple versions of a model. Model Compare from dSPACE can be used with any TargetLink, MathWorks® Simulink or Stateflow® model, and also supports libraries. There is a broad range of use cases for Model Compare, for example:

* Managing different model versions or model variants
* Merging parallel development branches
* Verifying and reviewing model changes

**TargetLink Support**

Model Compare has built-in support for all TargetLink blocks and properties. They are displayed and handled like ordinary Simulink/Stateflow elements, so you do not have to bother with TargetLink implementation details. Since the semantics of TargetLink properties are known, they are also intelligently handled by the predefined filter options.

### Review Sessions and Merge Support

You can associate review comments to block and property differences found by the tool, including date/author tracing. Complex reviews with multiple participants are supported.

Detected changes can be transferred from one model to another to merge parallel development branches or manage different model variants. With easy-to-use commands, merging models this way is much less error-prone than it would be by hand. The remaining differences between the models are constantly kept up-to-date, so that you always see the current state of your work.

### Interaction with Version Control Systems

Model Compare can be integrated in different version control systems (VCS), including Git and command-line interface-based version control. Furthermore, Model Compare can be used in your continuous integration (CI) pipeline, e.g., for generating diff reports for review purposes.

**Key Benefits**

Model Compare finds all the changes in a model. Even large models can be compared in minutes, which would be practically impossible without tool support. Furthermore, the optional three-way model analysis detects conflicting changes and shows how models have changed rather than only their differences. You can use filters to focus on relevant differences and ignore unimportant ones such as layout changes or simulation settings. The support for review sessions also enables reliable, safe, and controlled reviews of model changes. The merge support simplifies the synchronization of changes in different versions of a model.

**Activity 3: Comparing Model Accuracy Before & After Applying Hyperparameter Tunning**

### Application Areas

Automatic comparison is a must whenever you work with multiple versions of a model. Model Compare from dSPACE can be used with any TargetLink, MathWorks® Simulink or Stateflow® model, and also supports libraries. There is a broad range of use cases for Model Compare, for example:

### TargetLink Support

* Managing different model versions or model variants
* Verifying and reviewing model changes

Model Compare has built-in support for all TargetLink blocks and properties. They are displayed and handled like ordinary Simulink/Stateflow elements, so you do not have to bother with TargetLink implementation details. Since the semantics of TargetLink properties are known, they are also intelligently handled by the predefined filter options.

**Review Sessions and Merge Support**

You can associate review comments to block and property differences found by the tool, including date/author tracing. Complex reviews with multiple participants are supported.

Detected changes can be transferred from one model to another to merge parallel development branches or manage different model variants. With easy-to-use commands, merging models this way is much less error-prone than it would be by hand. The remaining differences between the models are constantly kept up-to-date, so that you always see the current state of your work.

### Interaction with Version Control Systems

Model Compare can be integrated in different version control systems (VCS), including Git and command-line interface-based version control. Furthermore, Model Compare can be used in your continuous integration (CI) pipeline, e.g., for generating diff reports for review purposes.

**Key Benefits**

Model Compare finds all the changes in a model. Even large models can be compared in minutes, which would be practically impossible without tool support. Furthermore, the optional three-way model analysis detects conflicting changes and shows how models have changed rather than only their differences. You can use filters to focus on relevant differences and ignore unimportant ones such as layout changes or simulation settings. The support for review sessions also enables reliable, safe, and controlled reviews of model changes. The merge support simplifies the synchronization of changes in different versions of a model.

**Milestone5:Performance Testing&Hyperparameter Turning**

**Activity 1: Save The Best Model**

**Activity 2: Integer With Web Framework**

**Activity 3: Building Html Pages**

**Activity 4: Build Python Code**

**Activity 5: Run The Web Application**

**Activity 1: Save The Best Model**

**Saving best model**

Asked 5 years, 3 months ago

Modified [1 month ago](https://stackoverflow.com/questions/48285129/saving-best-model-in-keras?lastactivity)

Viewed 84k times

65

I use the following code when training a model in keras

from keras.callbacks import EarlyStopping

model = Sequential()

model.add(Dense(100, activation='relu', input\_shape = input\_shape))

model.add(Dense(1))

model\_2.compile(optimizer='adam', loss='mean\_squared\_error', metrics=['accuracy'])

model.fit(X, y, epochs=15, validation\_split=0.4, callbacks=[early\_stopping\_monitor], verbose=False)

model.predict(X\_test)

but recently I wanted to get the best trained model saved as the data I am training on gives a lot of peaks in "high val\_loss vs epochs" graph and I want to use the best one possible yet from the model.

Is there any method or function to help with that?

* [python](https://stackoverflow.com/questions/tagged/python)
* [keras](https://stackoverflow.com/questions/tagged/keras)
* [deep-learning](https://stackoverflow.com/questions/tagged/deep-learning)
* [neural-network](https://stackoverflow.com/questions/tagged/neural-network)

Adding the code here in case the above Kaggle example link is not available:

model = getModel()

model.summary()

batch\_size = 32

earlyStopping=EarlyStopping(monitor='val\_loss', patience=10, verbose=0, mode='min')

* mcp\_save = Managing different model versions or model variants

ModelCheckpoint('.mdl\_wts.hdf5', save\_best\_only=True, monitor='val\_loss', mode='min')

reduce\_lr\_loss = ReduceLROnPlateau(monitor='val\_loss', factor=0.1, patience=7, verbose=1, epsilon=1e-4, mode='min')

model.fit(Xtr\_more, Ytr\_more, batch\_size=batch\_size, epochs=50, verbose=0, callbacks=[earlyStopping, mcp\_save, reduce\_lr\_loss], validation\_split=0.25)

[EarlyStopping](https://keras.io/callbacks/#earlystopping)'s restore\_best\_weights argument will do the trick:

**restore\_best\_weights:** whether to restore model weights from the epoch with the best value of the monitored quantity. If False, the model weights obtained at the last step of training are used.

So not sure how your early\_stopping\_monitor is defined, but going with all the default settings and seeing you already imported EarlyStopping you could do this:

early\_stopping\_monitor = EarlyStopping(

monitor='val\_loss',

min\_delta=0,

patience=0,

verbose=0,

mode='auto',

baseline=None,

restore\_best\_weights=True

], verbose=F)

And then just call model.fit() with callbacks=[early\_stopping\_monitor] like you already do.

* 1

Man... Keras is just *too* easy!

– [Ulf Aslak](https://stackoverflow.com/users/3986879/ulf-aslak)

[Oct 2, 2020 at 11:09](https://stackoverflow.com/questions/48285129/saving-best-model-in-keras#comment113474623_59383258)

* 1

will it still restore the best weights when it didn't manage to earlystop but finish all epochs

– [noone](https://stackoverflow.com/users/5229871/noone" \o "6,018 reputation)

[Jan 1, 2022 at 21:48](https://stackoverflow.com/questions/48285129/saving-best-model-in-keras#comment124716512_59383258)

* please note currently, restore\_best\_weights=False by default!

– [Ali Pardhan](https://stackoverflow.com/users/5728614/ali-pardhan)

[Apr 13, 2022 at 1:46](https://stackoverflow.com/questions/48285129/saving-best-model-in-keras#comment126969208_59383258)

[Add a comment](https://stackoverflow.com/questions/48285129/saving-best-model-in-keras)

15

I guess model\_2.compile was a typo. This should help if you want to save the best model w.r.t to the val\_losses -

checkpoint = ModelCheckpoint('model-{epoch:03d}-{acc:03f}-{val\_acc:03f}.h5',

verbose=1,

monitor='val\_loss',

save\_best\_only=True,

mode='auto'

)

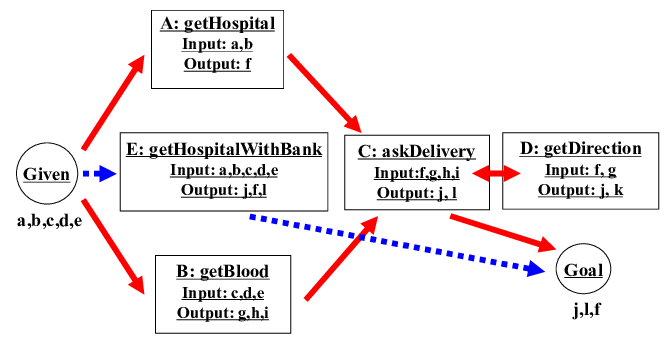
model.compile(optimizer='adam', loss='mean\_squared\_error', metrics=['accuracy'])

model.fit(X, y, epochs=15, validation\_split=0.4, callbacks=[checkpoint false)

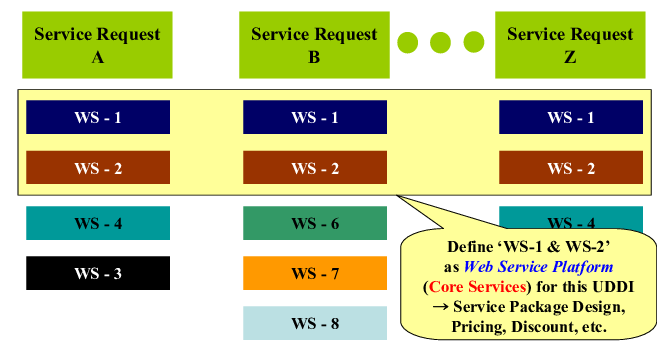
**Activity 2: Integer With Web Framework**

**Abstract and Figures**

In this paper, we propose a Web service composition framework that uses Integer Linear Programming with non-functional objectives and constraints, in addition to the syntactic matching of Web services features. We envision that when Web services are fully deployed and commercialized in the near future, the criteria of Web service composition to achieve objectives will vary depending on users' needs or preferences from the number of Web services to non-functional objectives, such as costs, time, and/or reputation. Such non-functional attributes cannot be readily considered in planning-graph, constraint satisfaction, or propositional satisfiability techniques, which are predominantly logic-based. This paper shows how the proposed Integer Linear Programming framework can be utilized to compose Web services with non-functional attributes. This framework enables our composition software agent to identify the best composition result that satisfies both non-functional requirements as well as functional ones, namely, parameter matching. A preliminary implementation of the proposed idea and further research directions are also discussed.

[](https://www.researchgate.net/figure/A-use-case-with-non-functional-attributes_fig1_221543040" \o "Figure 1. A use case with non-functional attributes  )

**[A use case with non-functional attributes](https://www.researchgate.net/figure/Web-service-consolidation_fig3_221543040" \o "Figure 5. Web service consolidation  )**



**Web service platform identification**

**Activity 3: Building Html Pages**

### [Introduction](https://www.digitalocean.com/community/tutorial_series/how-to-build-a-website-with-html#introduction)

If you are interested in learning how to build and design websites, Hyper Text Markup Language (HTML) is a great place to start. This project-based tutorial series will introduce you to HTML and its methods by building a personal website using our [demonstration site](https://html.sammy-codes.com/) (below) as a model. Once you learn the basics, you will know how change the website’s design and add personalized content. No prior coding experience is necessary to follow along the tutorials in this series.

## [Prerequisites](https://www.digitalocean.com/community/tutorial_series/how-to-build-a-website-with-html#prerequisites)

* A code editor like [Visual Studio Code](https://code.visualstudio.com/download) or [Atom](https://atom.io/). For this tutorial series, we will be using Visual Studio Code as our default code editor but you may use any code editor you like. Certain instructions may need to be slightly modified if you use a different editor.
* A web browser like [Firefox](https://www.mozilla.org/en-US/firefox/new/) or [Chrome](https://www.google.com/chrome/). We will be using Firefox as our default browser but you may use any browser you like. Certain instructions may need to be slightly modified if you use a different web browser.
* Two different profile photos, images, or avatars for personalizing your site (optional).

**[How To View the Source Code of an HTML Document](https://www.digitalocean.com/community/tutorials/how-to-view-the-source-code-of-an-html-document" \t ")**

* This tutorial will introduce you to a basic HTML document and teach you how to view the source code of an HTML document in a browser.
* HTML is used to mark up a document with instructions that tell a browser how to display and interpret the document’s content. For example, HTML can tell the browser which text content should be interpreted as a heading and which text content should be interpreted as paragraphs. HTML is also used to add images and assign links to text and images. These instructions are communicated through HTML tags, which are written like this: <tagname>. Many, though not all tags, use an opening tag and closing tag to wrap around the content that they are used to modify.
* To get a sense of how these tags are used, let’s inspect a snippet of HTML code. The HTML code below shows how HTML tags are used to structure text and add links and images. Don’t worry if you don’t understand the tags immediately- we’ll study those in the next tutorial.

<h1>Sammy's Sample HTML</h1>

<p>This code is an example of how HTML is written.</p>

<p>It uses HTML tags to structure the text.</p>

<p>It uses HTML to add a <a href="digitalocean.com/community">link</a>.</p>

<p>And it also uses HTML to add an image:</p>

<img src="https://html.sammy-codes.com/images/small-profile.jpeg"/>

**4: Build Python Code**

**Overview:**

* [Introduction](https://www.devdungeon.com/content/how-build-python-source#toc-1)
* [Build steps](https://www.devdungeon.com/content/how-build-python-source#toc-2)
* [Get the source code](https://www.devdungeon.com/content/how-build-python-source#toc-5)
* [Configure](https://www.devdungeon.com/content/how-build-python-source#toc-6)
* [Build](https://www.devdungeon.com/content/how-build-python-source#toc-7)
* [Post-build](https://www.devdungeon.com/content/how-build-python-source#toc-8)
* [Further reading](https://www.devdungeon.com/content/how-build-python-source#toc-3)

Installing Python is easy using the pre-built installers and packages from your operating system. However, if you want to build the cutting-edge version directly from GitHub master branch, you will have to build your own version from source. You may also want to do it just to reinforce your understanding of Python.

This guide will walk through the steps needed to build Python 3 from source and then create a virtual environment that you can use for projects.

## Build steps

This assumes a Linux development environment and was tested in Fedora 30. You may need to install some of the necessary dev tools like  or gcc make  on your system.

**The main steps are:**

* Obtain the source code
* Run the configure script
* Run make
* Run make install

### GET THE SOURCE C

### A configure script comes in the source that can be passed many values. This will create the Makefile. One in particular that is important is the --prefix. This will determine where the final built files go.

./configure --prefix=$HOME/python3.9

By default, it will generate (among other things) a libpython3.9.a static library that you can use to link with C applications (e.g. gcc my.c -lpython3.9). If you want to build the shared library instead of the static one (libpython3.so), the use the --enabled-shared flag. This will NOT build the static library and you will have to ensure the .so file is loadable when running Python which adds more complexity. I recommend avoiding the shared library unless you have a need for it.

./configure --enable-shared

If you want to add optimizations, also add on the --enable-optimizations flag.

./configure --enable-optimizations

An example with all the options might look like this:

./configure --enable-optimizations --prefix=$HOME/python3.9

### BUILD

Use the make tool to build the files followed by make install to put the final files in the location specified by configure's --prefix.

make # Do the bulk of compilation

Near the end of the output, you should see a success message and some suggestions on optional modules. In the output below it mentions that it could not build \_tkinter.

Python build finished successfully!

The necessary bits to build these optional modules were **not** found:

\_dbm \_gdbm \_tkinter

nis readline

To find the necessary bits, look **in** setup.py **in** detect\_modules() **for** the **module**'s name.

The following modules found by detect\_modules() in setup.py, have been

built by the Makefile instead, as configured by the Setup files:

\_abc atexit pwd

time

In Fedora, the optional missing dependencies can be installed with:

dnf install tk-devel readline-devel gdbm-devel

In Debian, the packages are:

apt install libncurses-dev libgdbm-dev libz-dev tk-dev libsqlite3-dev libreadline-dev liblzma-dev libffi-dev libssl-dev

Run ./configure and make again to rebuild with the dependencies.

After the build was successful, compile all the final files in to the target destination that was specified by --prefix on the configure script.

make install # Puts the final files in prefix location

### POST-BUILD

After it performs the install, the prefix directory specified in configure will contain the output, which should be four directories:

├── bin

├── include

├── lib

└── share

* The bin dir contains the all-important python3 executable.
* The include directory contains all the include files needed for Python dev, including the important Python.h used for writing C extensions. This is the directory you would add to gcc with the -I flag if compiling C applications with embedded Python.
* The lib directory has all the Python modules as well as the libpython3.9.a library. Add this library directory to gcc search path for libraries with -L and link to the lib with -lpython3.9.
* The share directory will contain the man pages. Can be read with man (e.g. man ./share/man/man1/python3.9.1).

You could run Python directly from the bin/python3 executable, or add bin/ to your PATH environment variable.

Since Python 3 comes with the virtual environment package, I suggest creating a new virtual environment from the freshly built Python.

# Create a virtual environment from the new python

$HOME/mycpython/bin/python3 -m venv $HOME/venv

Then you can activate the virtual environment and ensure everything looks good.

source $HOME/venv/bin/activate

which python

which pip

python --version

pip --version

## Further reading

To learn more about virtual environments, see my [Python Virtual Environments Tutorial](https://www.devdungeon.com/content/python-virtual-environments-tutorial).

To learn more about Python's import paths, see my [Python import, sys.path, and PYTHONPATH Tutorial](https://www.devdungeon.com/content/python-import-syspath-and-pythonpath-tutorial).

## ConclusionAfter reading this, you should be able to build Python 3 from source using make and create a virtual environment using your built Python.

**Activity 5: Run The Web Application**

## What is a Webapp?

### Definition

Loosely speaking, a **web application** is a program that runs on a computer with a web server.

**From Wikipedia**:

*A****web application****(or****web app****) is application software that runs on a web server, unlike computer-based software programs that are run locally on the operating system (OS) of the device. Web applications are accessed by the user through a web browser with an active network connection. These applications are programmed using a client–server model—the user ("client") is provided services through an off-site server that is hosted by a third-party. Examples of commonly-used web applications include: web-mail, online retail sales, online banking, and online auctions.*

**So what’s a web server? Wikipedia says:**

*A web server is computer software and underlying hardware that accepts requests and send responses via HTTP or its secure variant HTTPS, the network protocols created to distribute web contents (web pages, etc.) to client user agents.*

What does this mean? It means the client sends requests to the server and the server responds::

