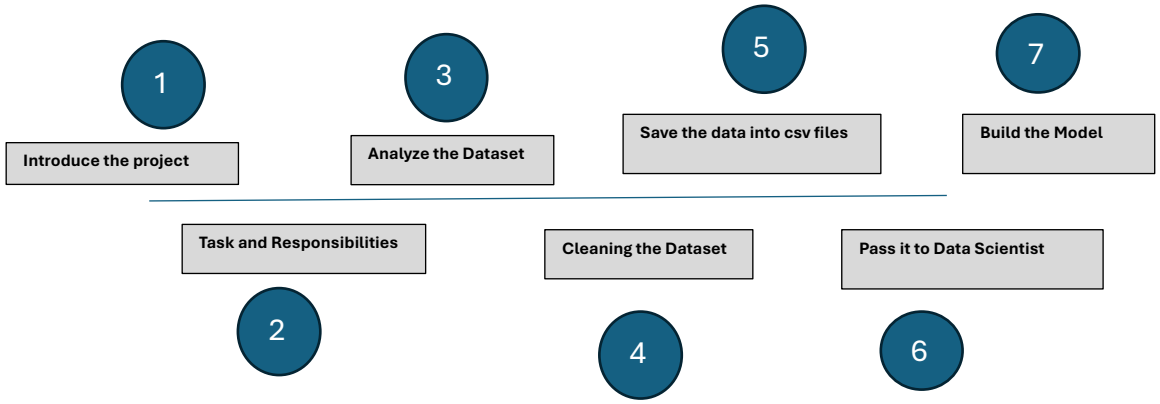
NumPy Project

Table of Contents

Pre-Steps Before Building the Model



Responsibilities Model

Data Analyst		Data Scientist
Gather Data Cleaning Preprocessing	→	Prepare Model
• Gather Data and prepare clean data set • Notedown All the changes • Create MarkDown README.md and explain each column and of data	→	Collect the Data documentation from Data Analyst Once the Each column is understood, start working on Model

Problem Statement

You are working in US in Federal bank, and you need to design credit risk model, and you need to design a model of finding the probability of default of every individual account. It has many steps such as **Recovery Rate**, **Probability of Default** and **Credit Risk Modelling**.

Task of Data Analysts:

- Gather the Dataset from the Federal bank
- Convert the US dollars into Euro
- Categorical Variable must be quantified
    - Convert text to numbers
- If the information is not available, assume the worst
    - Missing information is foul play
    - You need to be risk averse
- Casting Directions i.e. Choose MIN/MAX and AVG

DataSet

This project we, will be using the file called loan_data.csv:

- Pay attention to mix dataset such as date and string
- Delimiter
- Column header

A	B	C	D	E	F	G	H	I	J	K	L	M
id;issue_d;loan_amnt;loan_status;funded_amnt;term;int_rate;installment;grade;sub_grade;verification_status;url;addr_state;total_pymnt												
48010226;May-15;35000.0;Current;35000.0; 36 months;13.33;1184.86;C;C3;Verified;https://www.lendingclub.com/browse/loanDetail.action?loan_id=48010226;CA;9452.96												
57693261;;30000.0;Current;30000.0; 36 months; ¸ÎË.89;938.57;A;A5;Source Verified;https://www.lendingclub.com/browse/loanDetail.action?loan_id=57693261;NY;4679.7												
59432726;Sep-15;15000.0;Current;15000.0; 36 months;ÌÓÂ.53;494.86;B;B5;Verified;https://www.lendingclub.com/browse/loanDetail.action?loan_id=59432726;PA;1969.83												
53222800;Jul-15;9600.0;Current;9600.0; 36 months; ¸ÎË.89;300.35;A;A5;Not Verified;https://www.lendingclub.com/browse/loanDetail.action?loan_id=53222800;OH;1793.68												
57803010;Aug-15;8075.0;Current;8075.0; 36 months;19.19;296.78;;E3;Source Verified;https://www.lendingclub.com/browse/loanDetail.action?loan_id=57803010;TX;1178.51												

Steps to Pre-Process Data

Sno	Step	Code and Description
1	Import the numpy	
2	Before reading the data, set the np.set_printoptions so that you can clearly see the data such as	np.set_printoptions(suppress = True, linewidth = 100, precision = 2)
2	Load the data LO: try to load the file using loadtext and see the output LO: Pay attention to encoding,skip_header, and autostrip	raw_data_np = np.genfromtxt("loan-data.csv", delimiter = ';', skip_header = 1, autostrip = True,encoding = "cp855") raw_data_np
4	Check the incomplete data You can use function **isnan** to see the missing values.	np.isnan(raw_data_np).sum()
5	Let us use 2 variable temporary_fill and temporary_mean `Np.nanmean =>` Compute the arithmetic mean along the specified axis, ignoring NaNs. `Np.nanmax =>` Return the maximum of an array or maximum along an axis, ignoring any NaNs. When all-NaN slices are encountered a ``RuntimeWarning`` is raised and NaN is returned for that slice. In NumPy, axis=0 refers to the vertical axis, which runs downwards along the rows. When performing operations with axis=0, the operation is applied to each column, effectively aggregating data vertically. For a 2D array, axis=0 operates across the rows, while axis=1 operates across the columns. In higher-dimensional arrays, axis=0 corresponds to the first dimension. It is used in functions like sum, mean, max, etc., to specify the direction of the operation. Read: https://www.sharpsightlabs.com/blog/numpy-axes-explained/	temporary_fill = np.nanmax(raw_data_np) + 1 temporary_mean = np.nanmean(raw_data_np, axis = 0)

	If the array contains string, then it will be converted to Nan and np.nanmean might generate the warnings as the mean will be None and it means some column only contains the Nan value. However, the file which we are using is not the case because values are there, and string are converted to Nan.	
6	You can check temporary mean, and you should see nan values in total 8 columns. The column which are only Nan meaning they store only string.	
7	Generate temporary_stats with min,mean and max	temporary_stats = np.array([np.nanmin(raw_data_np, axis = 0), temporary_mean, np.nanmax(raw_data_np, axis = 0)])

Split the Dataset

Sno	Description	Status
1	Split the data to check if column is string and numeric Your job is to identify the column which full string and complete numeric You can use np.argwhere(np.isnan(temporary_mean)) == True => string np.argwhere(np.isnan(raw_data_np)) == False => Numeric	columns_strings = np.argwhere(np.isnan(temporary_mean)).squeeze() columns_strings columns_numeric = np.argwhere(np.isnan(temporary_mean) == False).squeeze() columns_numeric
2	Load the string data into string array. You need to use use_cols	loan_data_strings = np.genfromtxt("loan-data.csv", delimiter = ';', skip_header = 1, autostrip = True, usecols = columns_strings, dtype = str,encoding = "cp855") loan_data_strings
3	Load the string data into numeric array. You need to use_cols	loan_data_numeric = np.genfromtxt("loan-data.csv", delimiter = ';', autostrip = True, skip_header = 1, usecols = columns_numeric, filling_values = temporary_fill,encoding = "cp855") loan_data_numeric
4	Get the names of the column	header_full = np.genfromtxt("loan-data.csv", delimiter = ';', autostrip = True, skip_footer = raw_data_np.shape[0], dtype = np.str) header_full

5	Get the names of the column	header_full = np.genfromtxt("loan-data.csv", delimiter = ';', autostrip = True, skip_footer = raw_data_np.shape[0], dtype = str,encoding = "cp855") header_full
6	Separate the string and numeric columns	header_strings, header_numeric = header_full[columns_strings], header_full[columns_numeric]

Creating Checkpoints

Sno	Description	Status
1	Places where we store the copy of our dataset This is a very important technique as during the pre-processing and cleaning, we might lose the data. This is the fail safe to rely on.	def checkpoint(file_name, checkpoint_header, checkpoint_data): np.savez(file_name, header = checkpoint_header, data = checkpoint_data) checkpoint_variable = np.load(file_name + ".npz") return(checkpoint_variable)
2	Test the variable	checkpoint_test = checkpoint("checkpoint-test", header_strings, loan_data_strings)
3	Check the values	checkpoint_test['data']
	Use the equal function	np.array_equal(checkpoint_test['data'], loan_data_strings)

Manipulating String Columns

Sno	Description	Status
1	List the string	header_string
2	Set the correct column	header_strings[0] = "issue_date"
3	Check the strings	loan_data_strings

Date Handling

Sno	Description	Status
1	Check the date column	np.unique(loan_data_strings[:,0])

	Pay attention to date format 'MON-YY'	
2	Strip the year	loan_data_strings[:,0] = np.chararray.strip(loan_data_strings[:,0], "-15")
3	Check the strings	np.unique(loan_data_strings[:,0])
4	Convert the months to numeric value	for i in range(13): loan_data_strings[:,0] = np.where(loan_data_strings[:,0] == months[i], i, loan_data_strings[:,0])
5	Check the new value	np.unique(loan_data_strings[:,0])

Loan Status

Sno	Description	Status
1	List the header string	header_strings
2	List the value of first column	np.unique(loan_data_strings[:,1])
3	Check the unique columns	np.unique(loan_data_strings[:,1]).size
4	Convert the values to 0 or 1	loan_data_strings[:,1] = np.where(np.isin(loan_data_strings[:,1], status_bad),0,1)
5	Check the new value	np.unique(loan_data_strings[:,1])

Term

Sno	Description	Status
1	List the header string	header_strings
2	Get the unique terms	
3	Strip of the month	
4	Set the empty value default to 60 Use np.where	
5	Print the np.unique	

Grade and Subgrade

- Filling SubGrade
- Remove Grade
- Converting Subgrade

Verification Status

URL

State Address

Converting To Numbers

Checkpint1: String

Manipulating Numeric Columns

Substitute “Filler” values

ID

Temporary Stats

Funded Amount

Loaned Amount, Interest Rate, Total Payment, Installment