Loan Default Prediction with Machine Learning

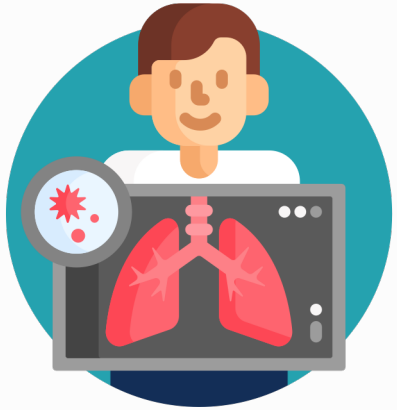
Course Introduction

Machine Learning solves real business problems.

Machine Learning models are used to make predictions and recommendations using data.

NETFLIX

Algorithms that make Netflix recommendations



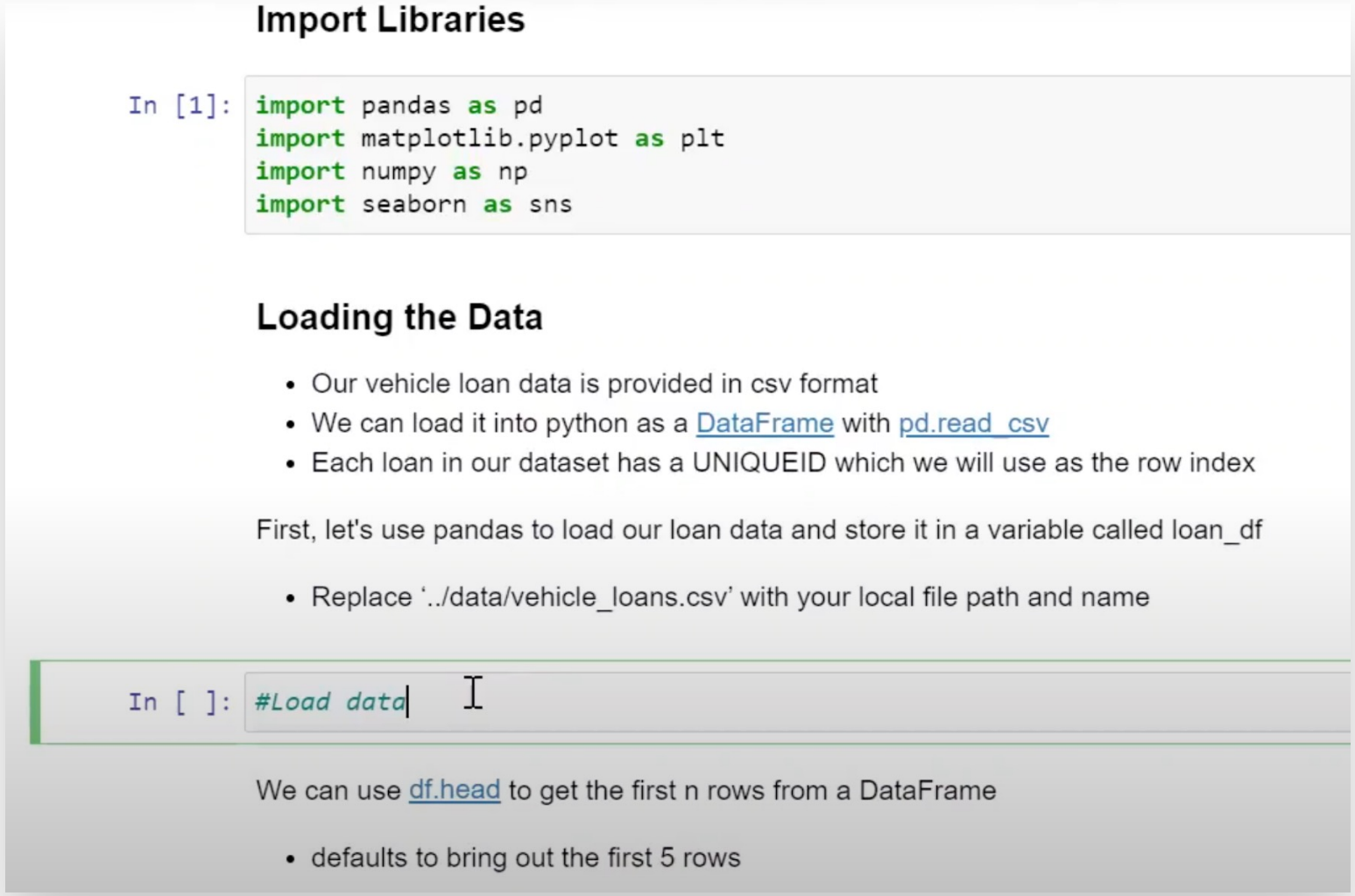
Models that use X-Rays to predict lung disease

Course Introduction

```
In [12]: #plot loan default
         sns.countplot(x="LOAN_DEFAULT", data=loan_df)
         plt.show()
```

- Build a classification model
- Predict if a customer is likely to default on their loans
- Inform risk calculations & pricing decisions

Course Introduction



Import Libraries

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
```

Loading the Data

- Our vehicle loan data is provided in csv format
- We can load it into python as a DataFrame with pd.read_csv
- Each loan in our dataset has a UNIQUEID which we will use as the row index

First, let's use pandas to load our loan data and store it in a variable called loan_df

- Replace '../data/vehicle_loans.csv' with your local file path and name

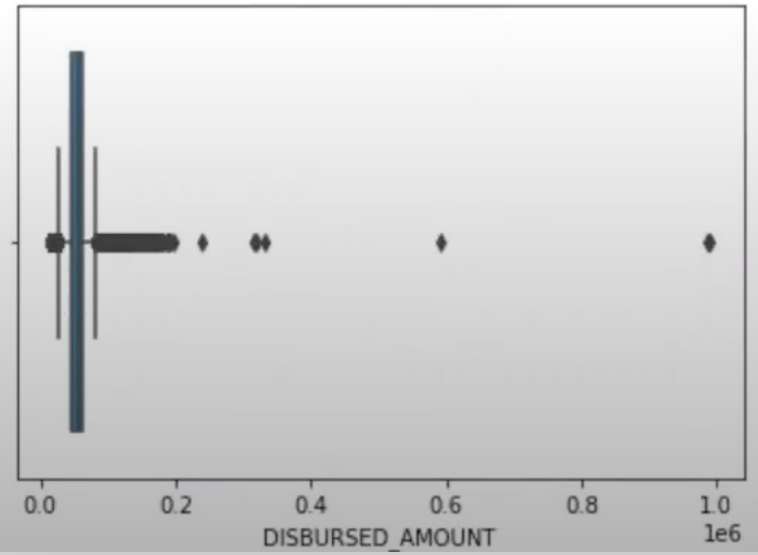
In [ ]:
```
#Load data
```

We can use df.head to get the first n rows from a DataFrame

- defaults to bring out the first 5 rows

Course Introduction

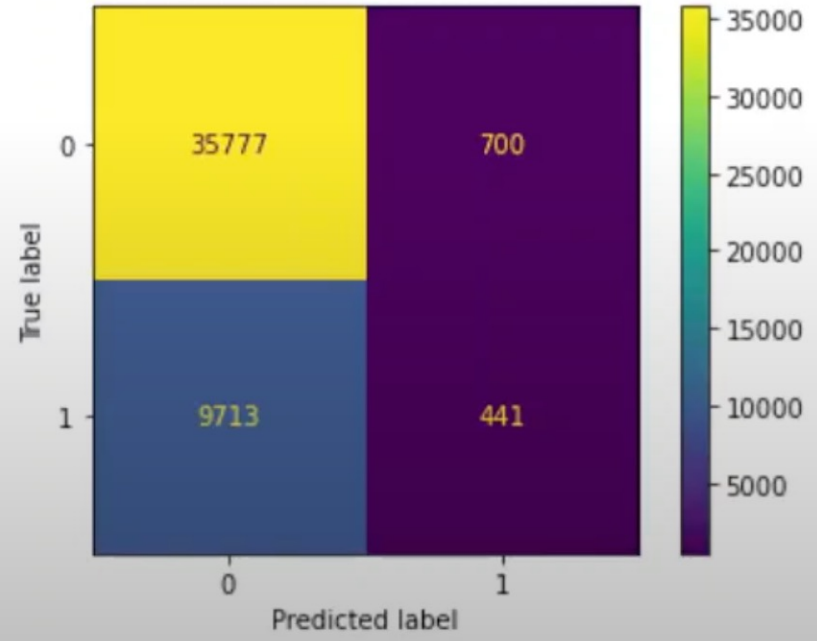
```
DISBURSED_AMOUNT Summary


count    233154.000000
mean      54356.993528
std       12971.314171
min       13320.000000
25%       47145.000000
50%       53803.000000
75%       60413.000000
max      990572.000000
Name: DISBURSED_AMOUNT, dtype: float64
```



Course Introduction

```
In [10]: #check the rows and columns
         print("x has {0} rows and {1} columns".format(x.shape[0], x.shape[1]))
         print("y has {0} rows".format(y.count()))

         x has 233154 rows and 5 columns
         y has 233154 rows

In [11]: #x info
         x.info()

         <class 'pandas.core.frame.DataFrame'>
         Int64Index: 233154 entries, 420825 to 630213
         Data columns (total 5 columns):
          #   Column             Non-Null Count   Dtype
         ---  ------             --------------   -----
          0   STATE_ID           233154 non-null  category
          1   LTV                233154 non-null  float64
          2   DISBURSED_CAT      233154 non-null  category
          3   PERFORM_CNS_SCORE  233154 non-null  float64
          4   DISBURSAL_MONTH    233154 non-null  category
         dtypes: category(3), float64(2)
         memory usage: 6.0 MB
```

```
In [11]: #type solution here
         rfc_model = RandomForestClassifier()
         rfc_model.fit(x_train, y_train)

         eval_model(rfc_model, x_test, y_test)
```



Course Introduction

```
In [ ]: import pandas as pd
        import matplotlib.pyplot as plt
        import numpy as np
        import seaborn as sns
        from sklearn.ensemble import RandomForestClassifier
        from sklearn.model_selection import train_test_split
        from sklearn.metrics import confusion_matrix, f1_score, accuracy_score, recall_score, roc_curve, auc, precision_score, plot_confu

In [ ]: loan_df = pd.read_csv('../data/vehicle_loans_feat.csv', index_col='UNIQUEID')
```

Chapter Overview

Load, clean, and explore the data set.

Use feature engineering to modify and improve data.

Build & evaluate logistic regression and random forest classification models.

Evaluate the accuracy of each model and explore ways to improve performance (i.e. class balancing).

This course is an end-to-end experience of how to implement a machine learning model, in python, from scratch.

What is Machine Learning

Machine learning (ML) is a technique that helps us make predictions by identifying patterns in data. 2 main types of machine learning:

Supervised ML

Helps us understand how a given variable is related to other variables in our dataset.

Start with a defined question- how do the loan & customer characteristics affect the likelihood of loan default? The model can then predict the likelihood of a default and classify loans accordingly.

Unsupervised ML

Start with a dataset of assumed real data, without a given variable in mind.

No specific question- the model reveals how the data is related.

What is Machine Learning

Machine learning (ML) is a technique that helps us make predictions by identifying patterns in data. 2 main types of machine learning:

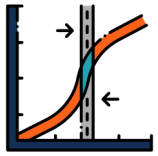
Supervised ML

Helps us understand how a given variable is related to other variables in our dataset.

Start with a defined question- how do the loan & customer characteristics affect the likelihood of loan default? The model can then predict the likelihood of a default and classify loans accordingly.

Binary Classification Model

Is a given loan going to default or not?



Logistic Regression

Random Forest Classifiers

Case Study Overview

Financial institutions incur significant losses due to loan defaults.

By modelling the risk of loan defaults, institutions can make more informed decisions about loan applications.

Customer	Risk
1	Low Risk
2	Low Risk
3	**High Risk**
4	Low Risk

Inform pricing decisions & price in risk

Calculate an informed risk evaluation of loans

Case Study Overview

This case will use a dataset of vehicle loans issued in India.

Customer	Credit Score	Loan Amount	Risk
1	700	500	Low Risk
2	800	7000	Low Risk
3	**300**	**5000**	**High Risk**
4	600	2000	Low Risk

Use:

- Customer credit scores
- Previous loan history
- Socio-economic factors

to predict if each customer will default on their loan.

Course Outline

Introduction

Load & Clean Data

Exploratory Data Analysis

Feature Engineering

Logistic Regression Classification

Model Evaluation

Random Forest Classification

Improving Accuracy

Load & Clean Data

Chapter Overview – Load & Clean Data

Load a CSV file of loan default data into a python notebook using pandas

Investigate the row and column counts

Identify columns that need cleaning

Identify and **deal with missing values**

Save cleaned data as a new CSV file

Manipulate data and prepare for exploration

Get to Know Your Data

- Before we go any further there are a few questions we need to answer about our data
  - How many data points do we have?
  - How many variables are there?
  - What are the variable types?
  - What timespan does the data cover?
  - Are there any missing values?
- Answering these basic questions provides context from which we can investigate further
- It will also help us identify individual columns which could need to be cleaned

DataFrames Theory

What is a DataFrame?

- Two-dimensional data structure made up from rows and columns
- Think about a table in Excel or SQL
- Rows represent data points and columns represent their attributes
- Columns have associated variable types (e.g. integer, string, date, list)

Why Dataframes?

- Conceptually easy to understand tabular data
- Pandas provides excellent tools for analyzing and manipulating dataframes

What is missing data?

- Many possible definitions
  - No information is stored for a particular variable in a row
  - Can be cells with ‘None’ or ‘Null’ values
  - Can also be cells with placeholder values e.g. ‘missing’

Types of Missing Data

- Missing Completely at Random (MCAR)
  - Fact that feature is missing is not related to some property of the data point
  - Caused by equipment failure, connection loss etc.
- Missing at Random (MAR)
  - Missingness of feature is not related to the feature itself but does have a relationship with other variables in the feature set
- Missing not at Random (MNAR)
  - Missingness of a feature has a direct relationship with the feature itself

Handling missing data

Why does it matter?

- Missing data can reduce the predictive power of a machine learning model
- Moreover, empty or null values can cause many algorithms to produce errors

How to deal with missing data?

- First step is to identify columns which have ‘NaN’, ‘NA’ or ‘None’ values and count their frequency
- Numerical missing values can be replaced with the mean or median value, depending on the distribution of the data
- Categorical missing values can be replaced with some placeholder category e.g. ‘Missing’ or filled with the most frequent value
- More advanced imputation techniques use algorithms such as K nearest neighbor clustering or linear regression to replace missing values

Dates and Strings

Dealing with dates

- Columns which store dates can not be included directly in machine learning models which perform mathematical operations/comparisons on features
- Date formatted columns often contain valuable information which can be preserved by converting the date into a numerical representation
  - E.g. We can calculate a person's age in years based on their date of birth

String Manipulation

- Some string columns contain useful categorical data
- However, they can also contain poorly formatted information which should be cleaned
  - E.g. Time period information stored in the format “1yrs 3mon”
- We can use string manipulation techniques such as splitting or regular expressions to identify patterns and extract data

Chapter Summary

- Basic use case for manipulating data for the ML project
- Identified target variable
- Dealt with missing data
- Calculated ages & extracted data from columns

Loading & Cleaning data is incredibly important, as it will make our data easier to work with.

Next up...  **Exploratory Data Analysis**

Exploratory Data Analysis

Chapter Overview – Exploratory Data Analysis

Pick variables for initial investigation

Create summary stats and plots

Check assumptions, spot anomalies, and form hypothesis

Create reusable functions in code, allowing you to repeat basic EDA far quicker next time around

Using EDA, our understanding of the data will help guide our model to the right variables.

Types of Features

There are **typically 4 types of features/columns:**

01

Unordered Categorical Features

Describe an item as being part of a specific group.

E.g. Employment Type- categorizes people into one of many groups.

These can be analyzed to see if they have an impact on the rate of loan defaults.

02

Ordered Categorical Features

Define buckets/groups that have some order between them.

E.g. Groups of house size- such as Small, Medium and Large- there is a clear order to these categories.

Types of Features

There are **typically 4 types of features/columns:**

03

Continuous Variables

Measurable variables such as asset costs and customer age.

The data shows numbers in the asset column are far bigger than those in the age column- needs to be fixed.

04

Binary Features

Define whether a certain condition is met. Also known as truth values or flags.

E.g. whether or not the loan applicant holds a passport.

1= true

0 = false

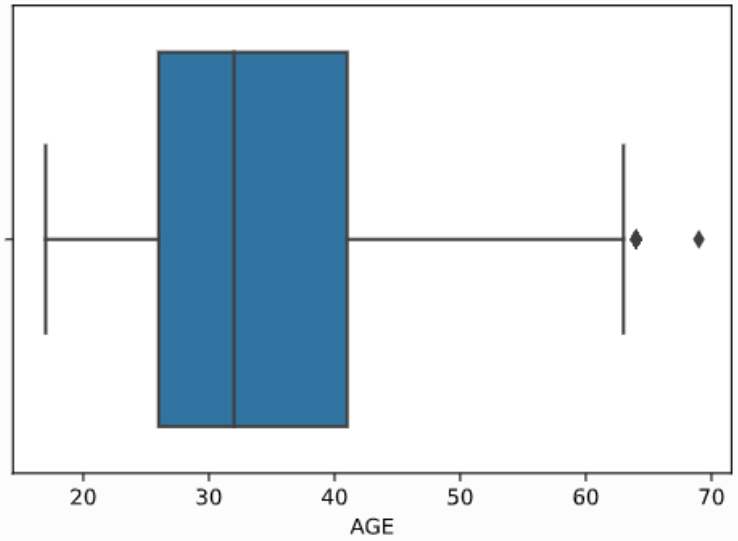
Summary Statistics

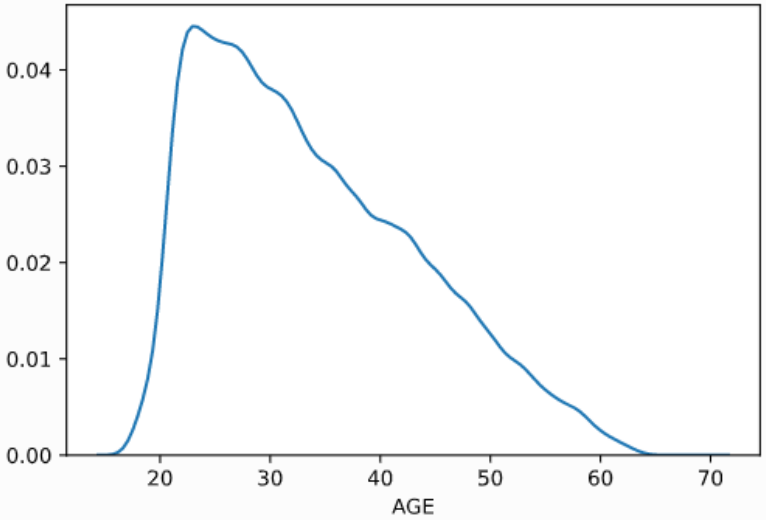
- Summary statistics for continuous variables are an essential part of exploratory data analysis
- Typically when we talk about summary statistic we are referring to
  - **Maximum -** Largest value
  - **Minimum -** Smallest Value
  - **Range –** Difference between the maximum and the minimum
  - **Mean -** Average Value
  - **Median -** Middle Value
  - **Standard Deviation -** How spread out is the data
  - **Inter Quartile Range (IQR) –** Range of the middle 50% of the data
- By generating summary statistics we can quickly get answers to some fundamental questions
  - How old is the average person in our data set?
  - What was the amount of the largest loan?
- Summary statistics can also help us spot errors
  - What does it mean if someone has an age of -1?

Boxplots and Distributions

Boxplots

- Provide a visualization of the summary statistics
- Minimum, maximum, Outliers, IQR and Median
- Allow us to confirm assumptions about skewness and extreme values
  - Age has some positive outliers



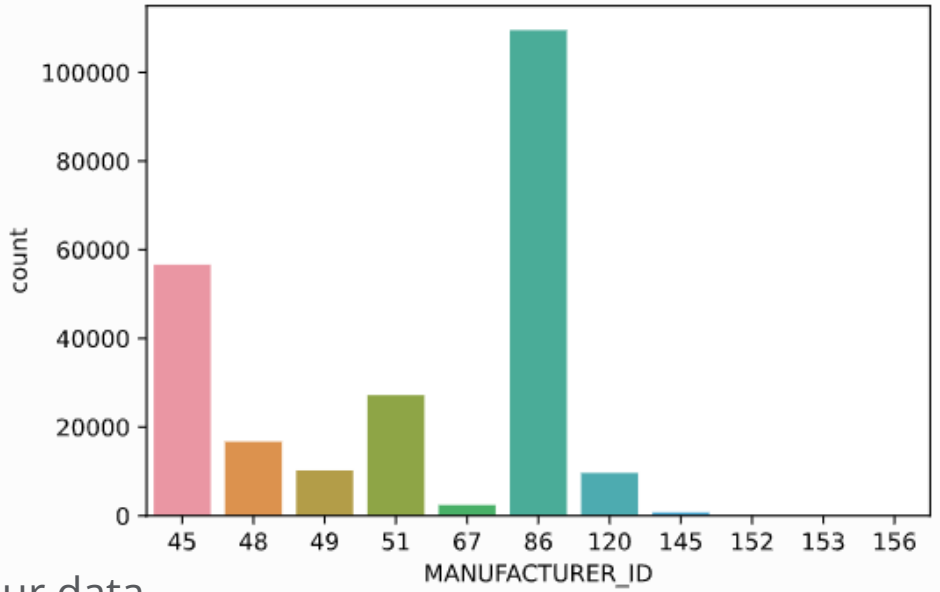


Distributions

- Show us the underlying frequency distribution of a variable
- The shape of the distribution gives us information about the data
  - Is it normally distributed?
  - Is it skewed?
  - How many peaks?

Bar Charts

- Very useful for visualizing categorical variables
- Show us the frequency of data for each category
- Essential part of EDA, we need to get an idea of common and uncommon values for our features



Hypothesis forming

- Using exploratory techniques we can start to form hypothesis about our data
- It is important to always keep in mind the relationship to the target variable
- Pandas and seaborn provide very useful functions that allow us to visualize this relationship and answer questions
  - Are loans for cars from certain manufacturers more likely to default?
  - Does the distribution of age change among people who defaulted on their loan?
  - What was the average value of loans which defaulted?

Chapter Summary

EDA is particularly important for supervised ML, where our model will benefit from some guidance as to which variables are most important.

- **Explored & plotted some of the most important features**
- **Understand how these features relate to loan defaults**
- **Familiarize with each type of feature**

Next up...  **Feature Engineering**

Feature Engineering

Chapter Overview – Feature Engineering

Make sure columns are in the optimal format for feature engineering

Learn how to use binning to create categorical variables from continuous ones

Create new features from existing ones and use scaling to normalize continues variables

Feature Engineering Techniques

Binning

- Create categorical variables from continuous data by grouping instances into bins and assigning labels to the bins
- Equal Width Binning
  - Split the continuous variables into bins with the same range/width
  - Bins will most likely have different numbers of data points
- Equal Frequency Binning
  - Split the continuous variables into bins of equal frequency
  - Each category has the same number of data points

More Feature Engineering

- We can create new features by combining existing ones
  - Create a new numeric column based on the sum of some existing columns
  - Create a new numeric column based on the ratio of some existing columns
  - Create a new binary column based on the value of some existing feature

Feature Scaling

- Majority of Data Science projects will encounter variables of vastly different ranges, magnitudes and units
- Depending on the choice of algorithm, models can end up weighted heavily towards variables with high magnitudes
- Imagine you have two columns describing different weights, with one in kilograms and another in grams.
  - Taken at face value some models will place more power towards the weight in grams
- Feature scaling aims to reduce this effect by bringing features into the same level of magnitude

Scaling Techniques

- Many different scaling techniques exist, two common approaches are:
  - **Min-Max Scaling -** rescale features into a range between 0 and 1
  - **Standardization -** rescale features to have a mean of 0 and standard deviation of 1. Essentially make features unitless

Chapter Summary

Feature engineering is important for modifying the structure of the input data to help our models be as effective as possible. Examples include:

- **Binning**
- **New columns**
- **Percentages**
- **Null Values & Scaling**

Next up...  **Classification with Logistic Regression**

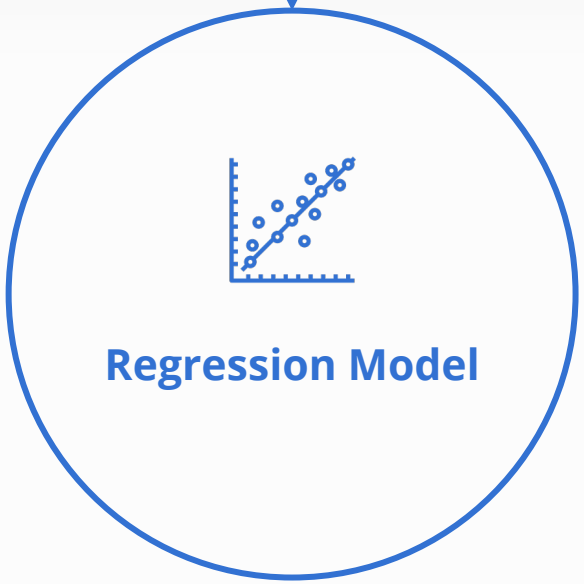
Classification with Logistic Regression

CFI

Chapter Introduction - Classification with Logistic Regression

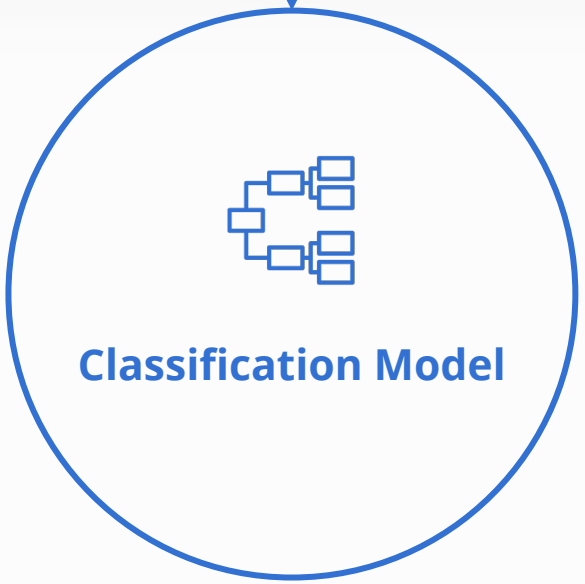
Supervised machine learning models fall into 2 categories:

Predict the value of a continuous variable



Regression Model

Predict a categorical variable

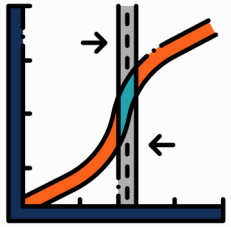


Classification Model

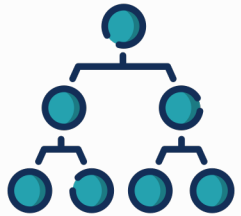
Binary Classification Model

Binary Classification Models

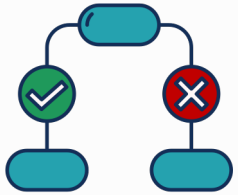
Some common examples of binary classification models include:



Logistic Regression



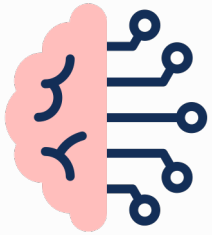
Decision Trees



Random Forest

Naïve Bayes

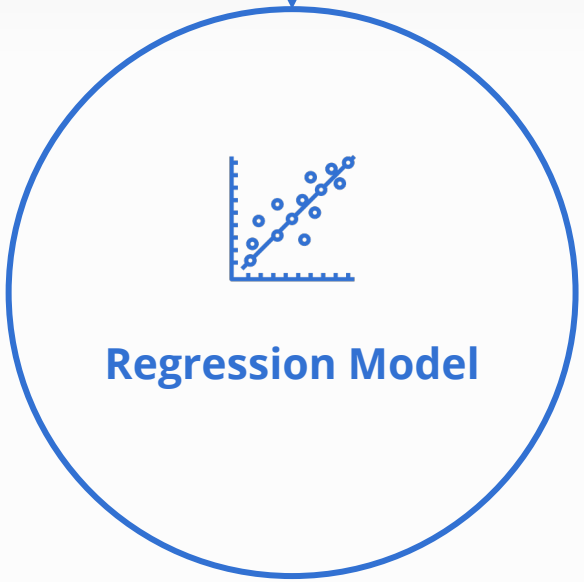
Nearest Neighbor



Neural Network

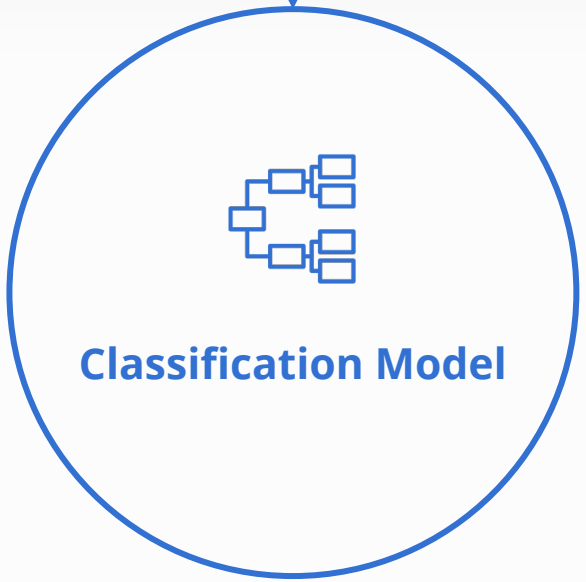
Linear Regression vs. Logistic Regression

Predict the value of a continuous variable



Regression Model

Predict a categorical variable

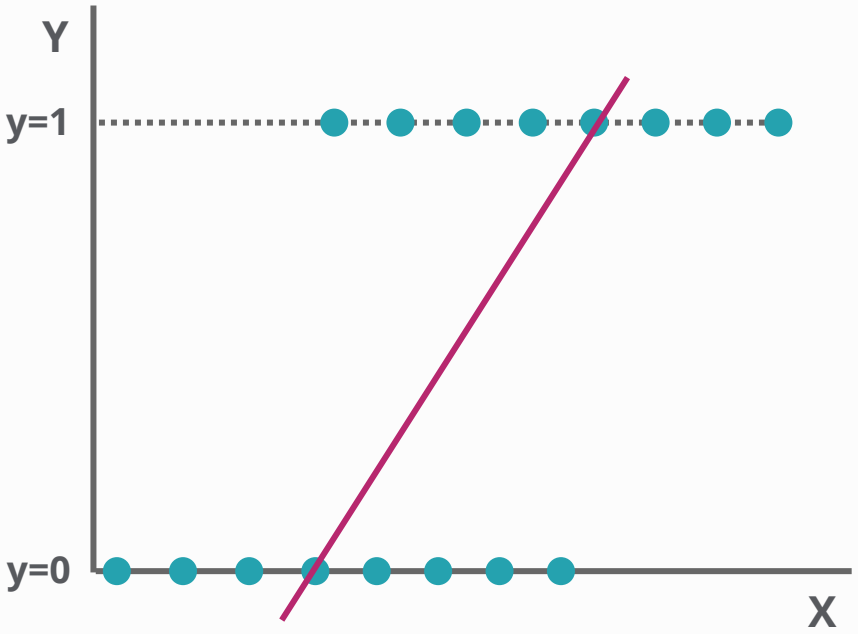


Classification Model

Binary Classification Model

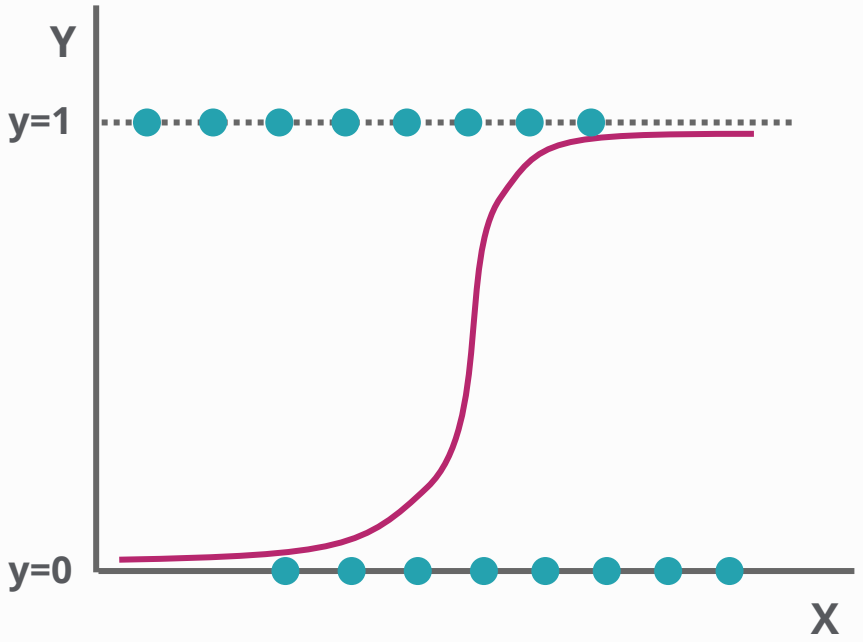
Linear Regression vs. Logistic Regression

The goal of a linear regression is to predict the value of an output variable (can take on any value), given one or more input variables.



Predicted Y can exceed 0 and 1 range.

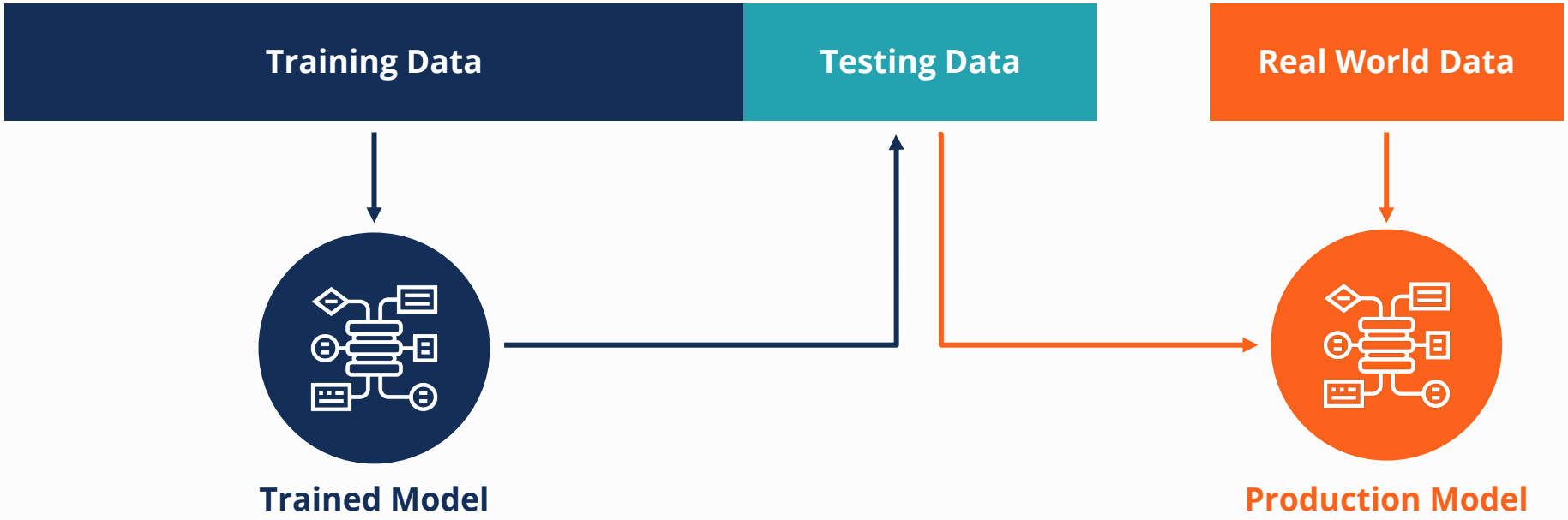
A logistic regression models an s-curve that only returns values between 0 and 1.



Predicted Y lies within 0 and 1 range.

Train & Test Split

Models need to be tested before we use them to predict real-world outcomes.



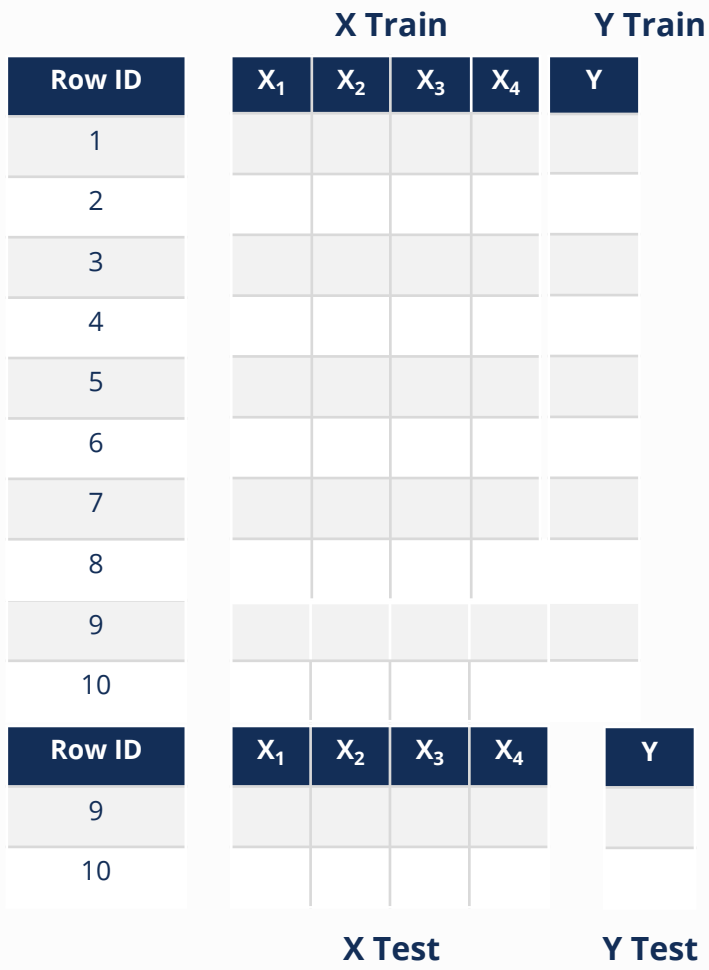
Train & Test Split

Splitting training and test data in Python involves 2 steps:

01 Split data frame into **inputs = x**, and **outputs = y**.

02 Split both x and y into training and test data sets.

Training = 0.8 | **Test size = 0.2**



Train & Test Split

In Python, sklearn includes the function **train_test_split** which allows us **to split up the data frame into training and test datasets.**

train_test_split = (X, Y, test_size, [random_state])

- **X, Y** — **X and Y dataframes**
- **test_size** — **Test size (0.2 in example)**
- **[random_state]** — **Dummy variables**

We need to reserve a portion of our data to test our model in order to see how well it performs on data it has not seen before.

Dummy Variables

Dummy variables help us turn unordered categorical features into useful columns of data.

Row ID	State ID
1	AB
2	NY
3	NJ
4	WS
5	OK
6	OH
7	HA
8	NX

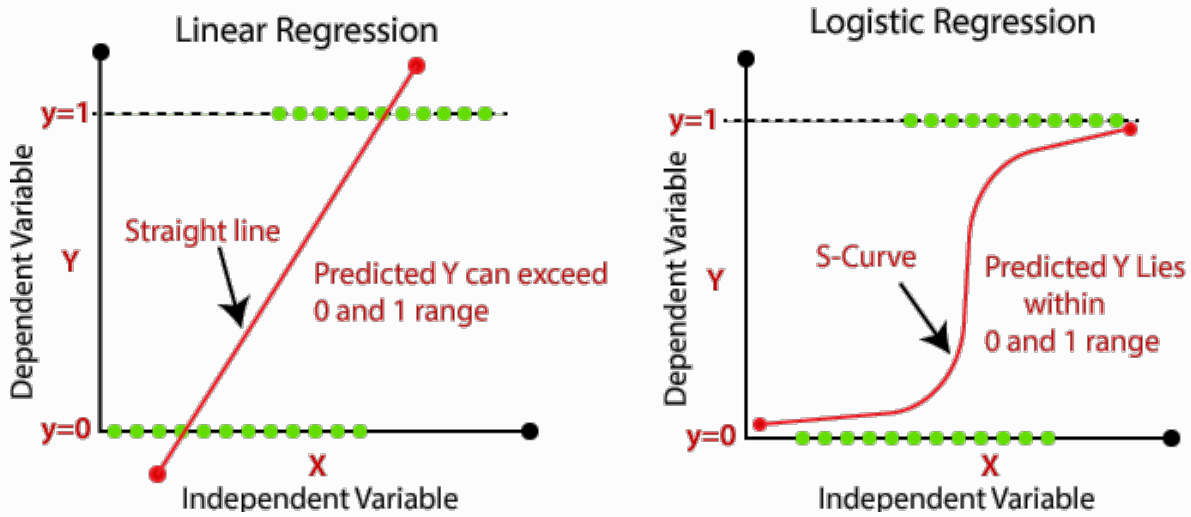
Row ID	AB	NY	NJ	WS	OK	OH	HA	NX
1	1	0	0	0	0	0	0	0
2	0	1	0	0	0	0	0	0
3	0	0	1	0	0	0	0	0
4	0	0	0	1	0	0	0	0
5	0	0	0	0	1	0	0	0
6	0	0	0	0	0	1	0	0
7	0	0	0	0	0	0	1	0
8	0	0	0	0	0	0	0	1

Logistic regression can't interpret string data.

One-hot-encoding: These columns can be used like any other binary column to determine if they affect the results.

Logistic Regression Theory

- Linear regression can be used to predict a continuous target variable
- How about our binary classification problem?



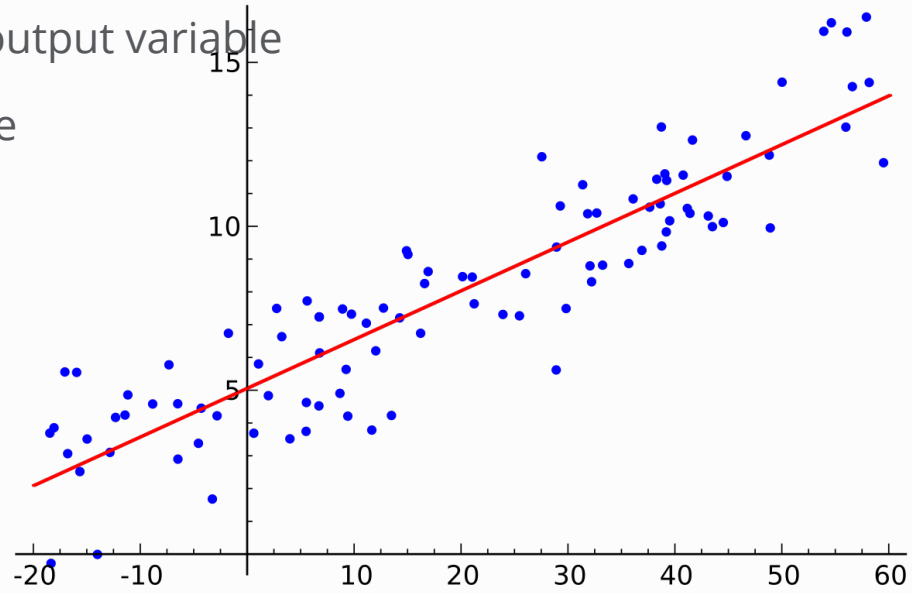
Logistic Regression

- Popular linear model that can be applied to binary classification problems
- Logistic regression uses an S-shaped logistic function to output a probability value between 0 and 1
- A data point can be classified by putting it through the function and classifying it according to a threshold
  - E.g. if the output probability is > 0.5 it belongs to class 1
- The function is fitted to the training data using a concept called maximum likelihood
  - Simply put, choose the line that maximizes the likelihood of observing the training data

Linear Regression Theory

- The goal of linear regression is to predict the value of a continuous output variable
- For example, we might use linear regression to answer questions like
  - What will the temperature be tomorrow?
  - How much will this house sell for?
  - How old do we think this person is?

$$age = \theta_0 + \theta_1 height + \theta_2 weight$$



- We can answer these questions by creating a linear function based on a weighted combination of our input variables
- Variable weights determine how much influence a particular variable has over the predicted output

Training Linear Regression

- How do we calculate the variable weights?
- Draw a line through the data using variable weights which minimize the Mean Squared Error (MSE) for our training data
  - MSE is the sum of the squared of distance between the predicted output and the actual value
- Training Data?
  - In supervised machine learning we train our models on a labelled subset of the total data
  - We will cover this in more detail later in the chapter
- Once the weights are settled we can use the model to predict our target variable for previously unseen data points

$$age = \theta_0 + \theta_1 height + \theta_2 weight$$

Training, Validation and Test Data

- In machine learning, we are trying to train a model to predict an output variable

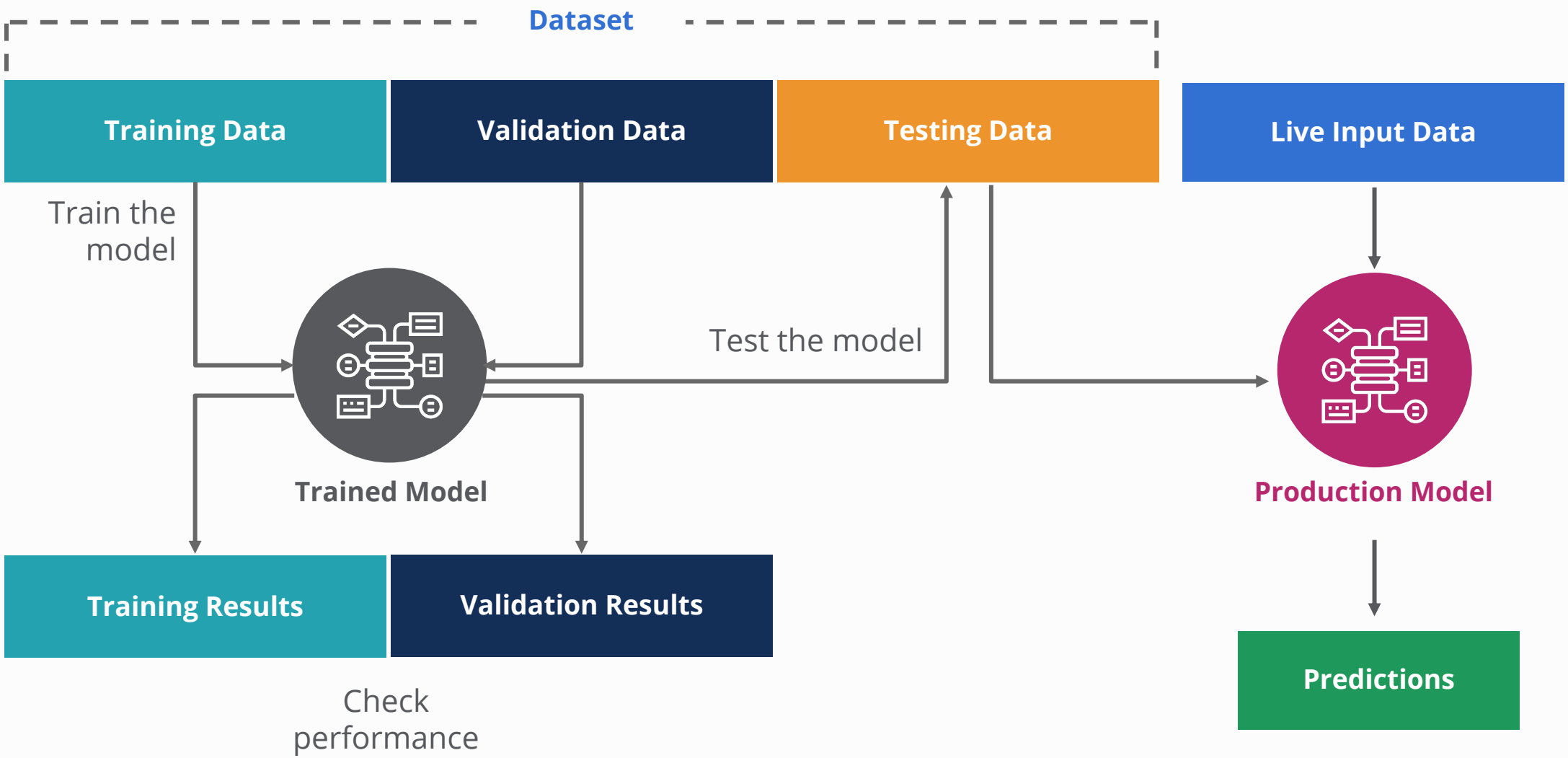
How do we build a model specific to our data set?

- Machine learning algorithms make predictions by adjusting certain values based on the data they are shown
  - Think about the weights in linear regression
- We need to show the model a labelled subset of our data so it can develop the ability to distinguish between one class and another

How do we test the model?

- The model should always be tested on previously unseen data
- To do this we need to split the data into training and test sets
  - **Training Data:** used to fit the model to our specific data
  - **Test Data:** used to test the predictive power of the trained model
- In some cases we may also create **Validation Data** which is used to fine-tune the model before testing

Train, Validation, and Test Split



Stratification

- It is important that class distribution in the test set matches the natural distribution within the data
- This must be kept in mind when splitting the data into training and test sets
- Imagine if the test data had no cases of loan default?
  - A model which classifies everything as a non-default would appear to perform well
- Similarly, what if the training data had no cases of loan default?
  - The model would be unable to learn any information about defaulted loans

Stratification

- The process of sampling the data to match the distribution of a certain variable
- Particularly useful for classification problems where the classes are unevenly distributed
  - Ensure that the class distribution in the test data is representative of the natural class distribution

Variable Encoding Theory

- Most machine learning algorithms cannot handle categorical/string values
- Variable encoding techniques can be used to create numeric representations of categorical data
  - Should always be before splitting data into training/test sets

One Hot Encoding

- Create a new 'dummy' variable for each category
- New variables are binary columns with 1 or 0 depending on the presence of the corresponding category

Integer/Label Encoding

- Replace categories with a number from 1 to n.
- n is the number of categories in the variable

Count/Frequency Encoding

- Count the number of observations in each category
- Replace the categories with this count

Ordered Integer Encoding

- Again, replace categories with a number from 1 to n.
- The number assigned to a category depends on the mean value of the target variable in that category

Chapter Summary

Congratulations! You have built a basic logistic regression model, which allows the user to predict whether loans are going to default or not.

We split data into X and Y dataframes, before using the train_test_split function to feed the logistic regression with the training data set.

Logistic regression is an essential part of your data science skill set.

Next up...  **Model Evaluation**

Model Evaluation

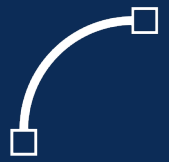
CFI

Chapter Overview – Model Evaluation

Familiarize with several metrics, charts and techniques that will help decide if the model is performing well or not.

These metrics, charts, and techniques can give a good indication of the predictive power of the model, whether it should be used for live data, and assist with comparisons between models.

Learn and use the formulas for Accuracy, Precision, Recall, and F1 score.

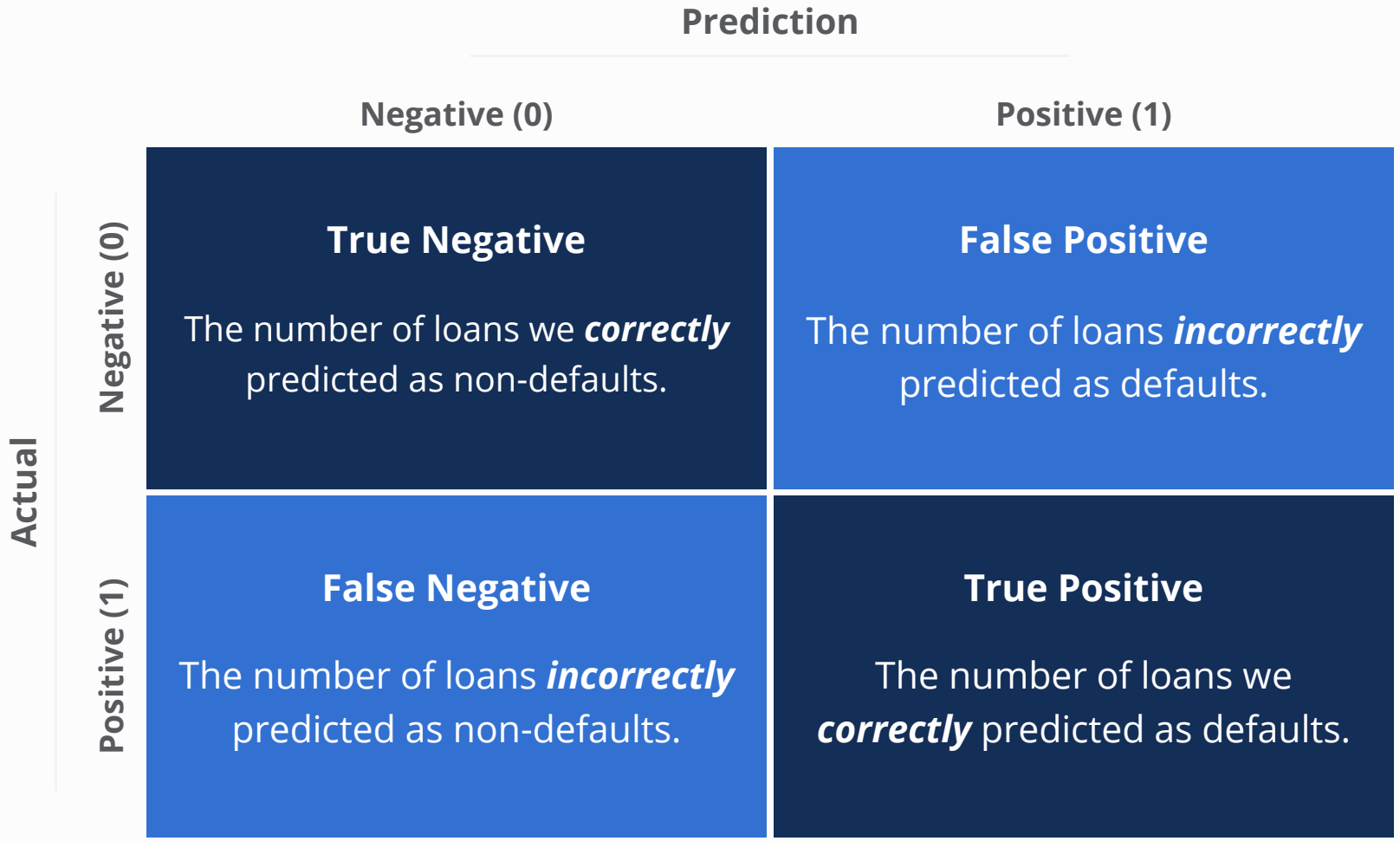


Plot the ROC curve & confusion matrices as additional ways to interpret/model results.

Explore more advanced evaluation techniques.

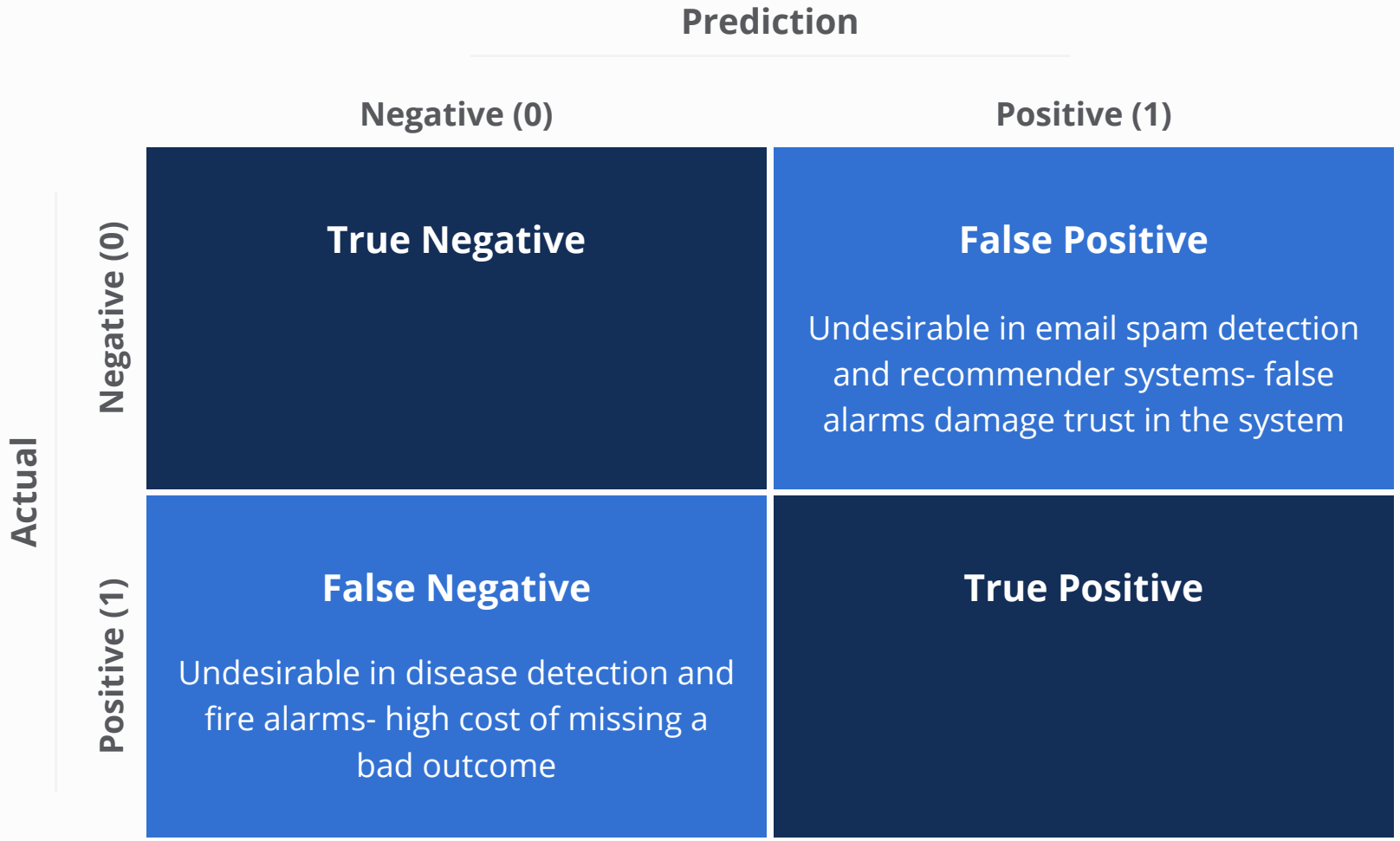
Confusion Matrix

The confusion matrix helps us understand the quality of our predictions.



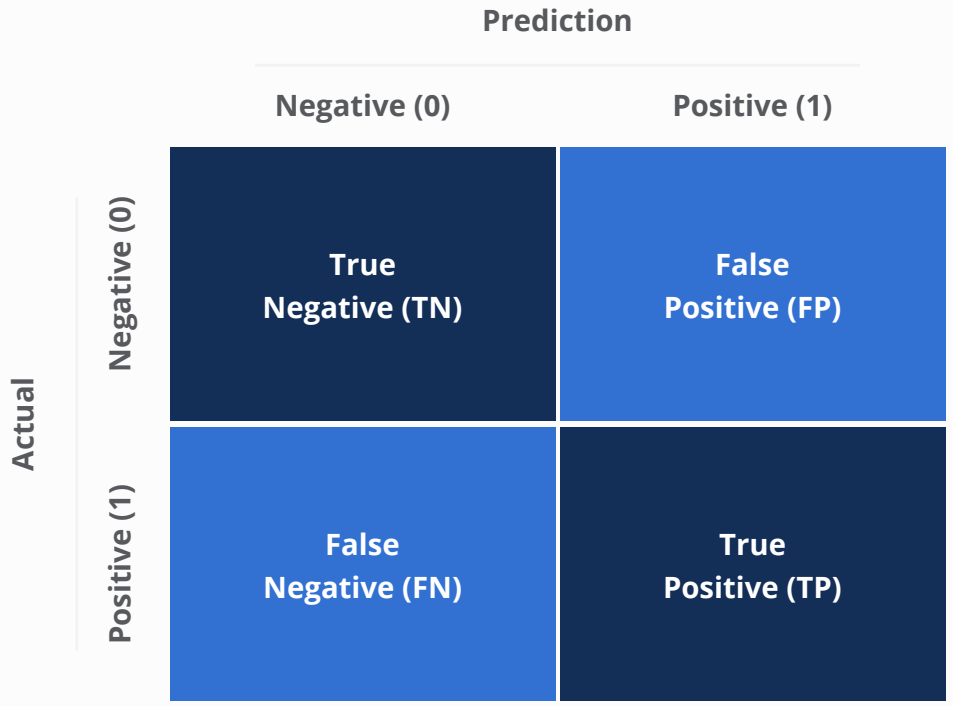
Confusion Matrix

Depending on the scenario, the confusion matrix can help us fine tune our model in different ways. For example:



Evaluation Metrics

There are **four key metrics** that can help summarize the observations in the confusion matrix:



1

Accuracy = (TN + TP) / Total Predictions

Describes what proportion of predictions were correct (may not always be the best indicator of performance).

2

Precision = TP / (TP + FP)

How good are the positive predictions? Out of those predicted positive, how many were actually positive?

3

Recall = TP / (TP + FN)

Describes what proportion of the actual positive cases were correctly identified.

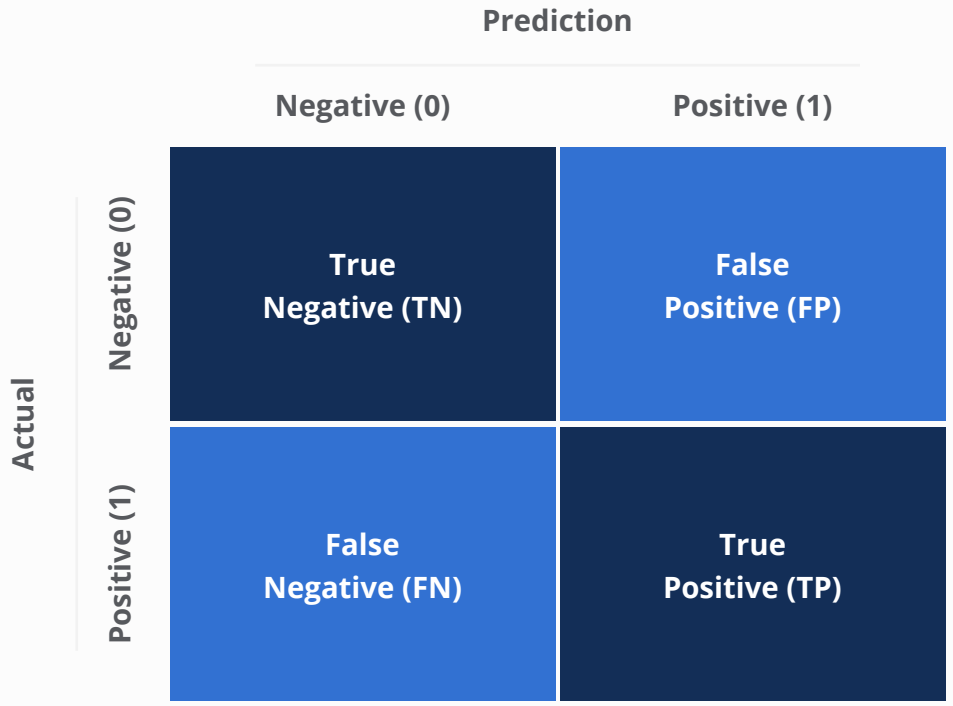
4

F1 Score = 2 * [(Precision*Recall) / (Precision+Recall)]

Provides a balance between precision and recall.

Evaluation Metrics

There are **four key metrics** that can help summarize the observations in the confusion matrix:



1

Accuracy = (TN + TP) / Total Predictions

Describes what proportion of predictions were correct (may not always be the best indicator of performance).

2

Precision = TP / (TP + FP)

How good are the positive predictions? Out of those predicted positive, how many were actually positive?

3

Recall = TP / (TP + FN)

Describes what proportion of the actual positive cases were correctly identified.

4

F1 Score = 2 * [(Precision*Recall) / (Precision+Recall)]

Provides a balance between precision and recall.

The ROC Curve

The Receiver Operating Characteristic (ROC) curve visualizes the model performance and is a useful way to evaluate the results of a binary classification model.

The ROC relies on 2 underlying **prediction percentages:**

True Positive Rate (TPR)

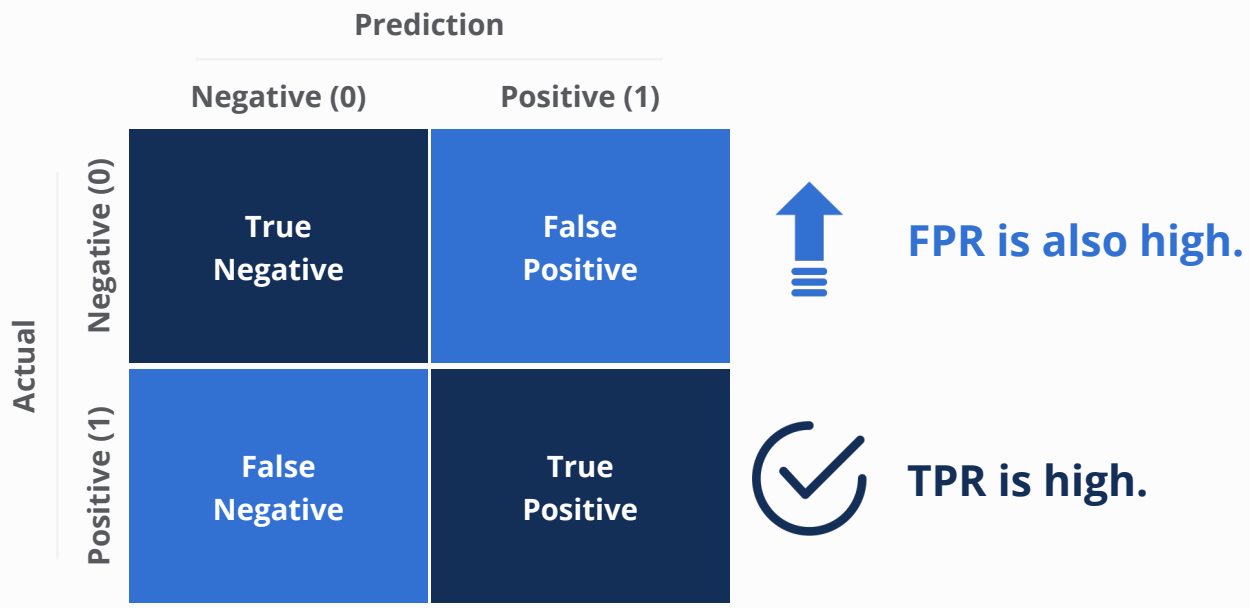
Describes the proportion of **actual positives** that we correctly identified. **Same as recall.**

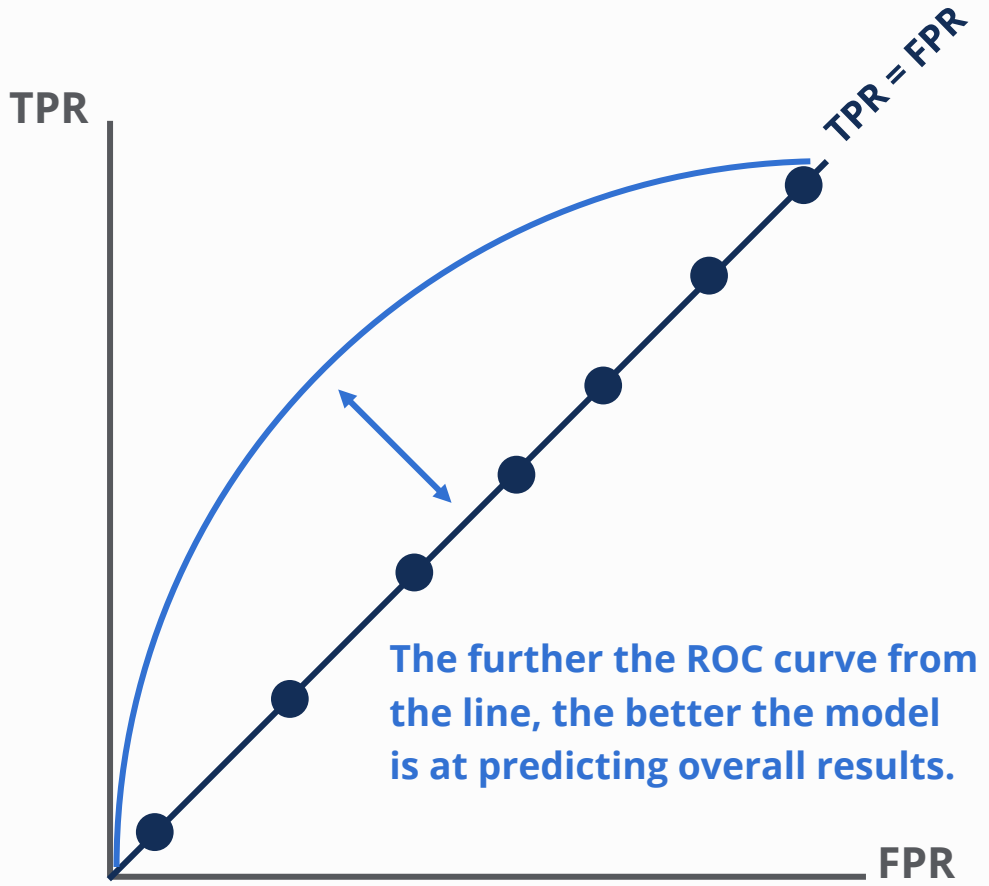
False Positive Rate (FPR)

Describes the proportion of **actual negatives** that we flagged as positive.

The ROC Curve

Suppose that for each loan, we guess default or non-default. **Letting random chance dictate our answers will result in a TPR very similar to FPR.**





Accuracy

- At the end of chapter 5 we introduced accuracy
- The simplest and most intuitive metric for classification problems
- Accuracy tells us the percentage of data points our model classified correctly

Accuracy can be misleading!

- Why would we care about anything except accuracy?
  - Imagine building a model for disease classification where only 1% of patients had the disease
  - A model that predicts no one ever has the disease would be 99% accurate
  - But would it be a good model?

Precision, Recall and F1

- We know that accuracy is not always the best measure
  - What other metrics should we consider?

Precision

- Out of those we predicted were positive, how many actually were positive?
- Useful when the cost of false positives is high. i.e. in an email spam filter

$$Precision = \frac{TP}{TP + FP}$$

Recall

- How many of the actual positive cases did we correctly identify?
- Useful when the cost of false negatives is high. i.e. in disease detection

$$Recall = \frac{TP}{TP + FN}$$

F1 Score

- Harmonic mean of precision and recall
- Useful when we need a balance between precision and recall
- Less affected by large numbers of true negatives than accuracy

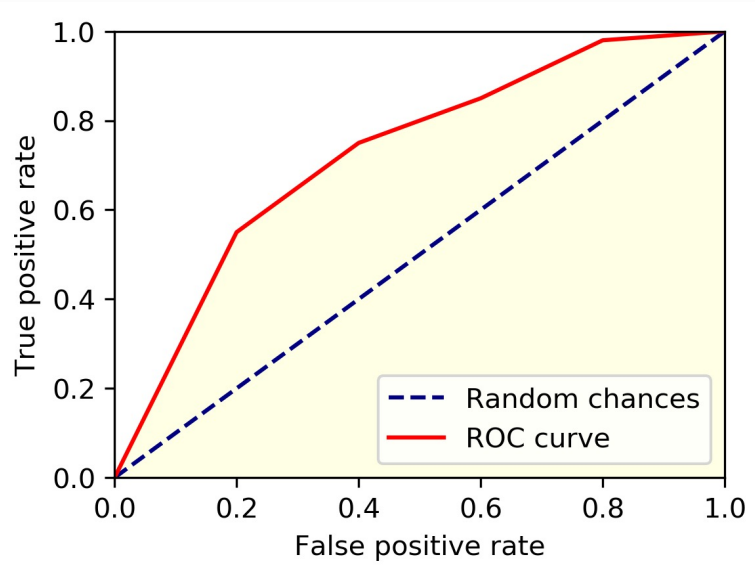
$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

What is the ROC Curve?

- So far the metrics we have looked at provide us with a very high-level view of model performance
- Usually we need a bit more detail, especially when optimising

Receiver Operating Characteristics (ROC) Curve

- The ROC curve is a probability curve
- It plots the True Positive Rate (TPR) on the y-axis
- False Positive Rate (FPR) on the x-axis



$$TPR = \frac{TP}{TP + FN} \qquad FPR = \frac{FP}{TN + FP} \qquad TPR = Recall = \frac{TP}{TP + FN}$$

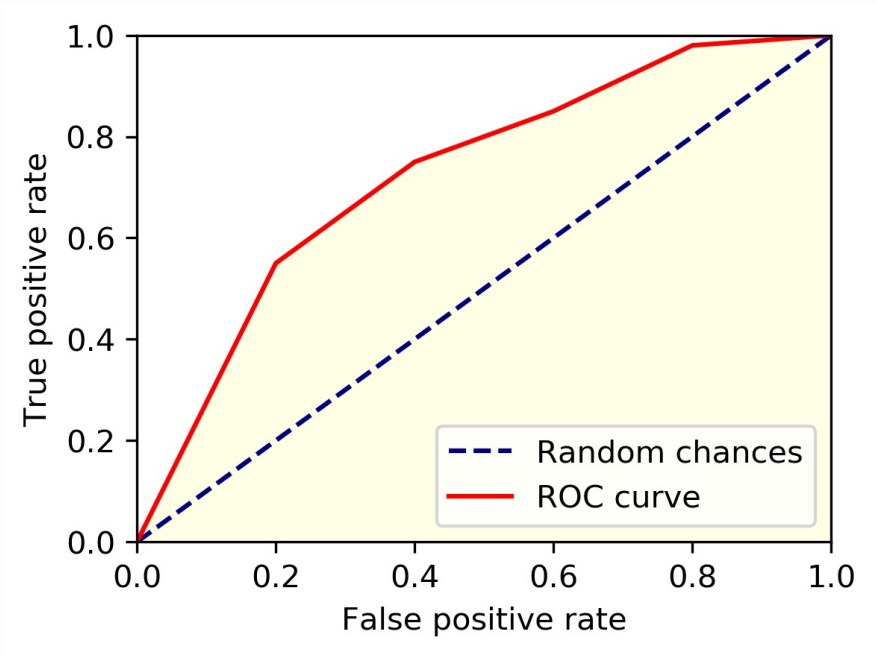
- Notice that TPR and Recall are the same
- To plot the ROC curve, we calculate the TPR and FPR for different probability thresholds between 0 and 1
  - The threshold is the probability at which an instance is classified in the positive class
  - With a threshold of 0.5, all instances > 0.5 would be classified as positive

Interpreting the ROC Curve

- The straight line plotted up the middle represents the performance of a random chance classifier

Area Under the Curve (AUC)

- The real power of the ROC curve comes from calculating the AUC
- Essentially this gives us a measure of separation
- In other words, how good is our model at predicting 1s as 1s and 0s as 0s
- The higher the AUC the better the model is at separating instances of the target variable
- The random classifier will have an AUC of 0.5



Advanced Model Evaluation

- Analysing the area under the ROC curve gives us a good indication of our models ability to separate classes
- Sometimes we might need to dig a little deeper, especially if the model is not performing well

What can we do?

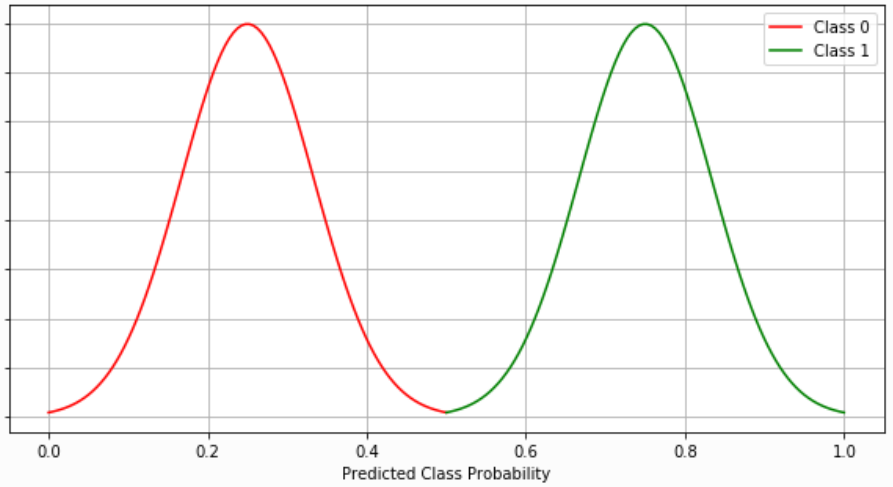
- We can expand on the confusion matrix idea by looking at the percentage classification splits
  - What percentage of our 1s were predicted as 1s (TPR)
  - What percentage of our 1s were predicted as 0s (FNR)
  - What percentage of our 0s were predicted as 0s (TNR)
  - What percentage of our 0s were predicted as 1s (FPR)
- This will give us insight into the where the problems with our model might lie

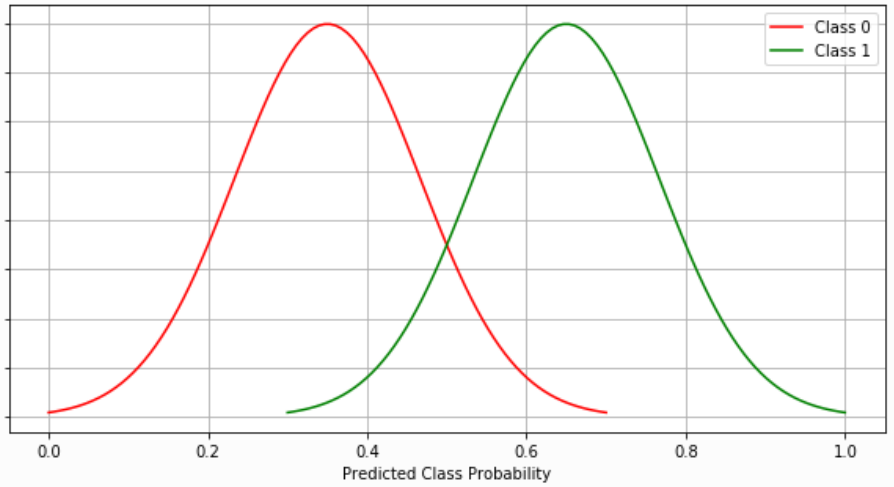
Class Probability Distributions

- Sklearn provides us with a method for extracting the target variable probabilities from our model
  - In other words, we have a list of values between 0 and 1 that tell us how likely the model thinks it is that a loan belongs to the default class
- We can visualize this by splitting the test data into groups according to the true class labels and plotting the predicted probability distributions for both groups
- In a perfect world these two distributions would meet at the classification threshold (0.5)
  - Instances of class 0 would have a predicted probability < 0.5 of belonging to class 1
  - Instances of class 1 would have a predicted probability > 0.5 of belonging to class 1

Class Probability Distributions

- Reality is not perfect!
  - These plots give us another way of examining the models ability to separate the two classes





Chapter Summary

Model performance is not an exact science as it relies on some subjective input from the user to determine what good performance looks like.

We have to **decide what errors we want to minimize** using the tools:

Confusion Matrix

Evaluation Metrics

ROC Curve

Prediction Percentages

Model evaluation is vital to the overall ML process.

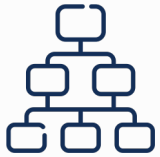
Next up...  **Random Forest Classification**

Classification with Random Forest

CFI

Chapter Introduction

The random forest model is considered an **ensemble model as its results are based on the combined output of multiple decision trees.**



Learn the basic theory on decision trees & random forests, before building a random forest classification model.

Create functions and review results of the model.

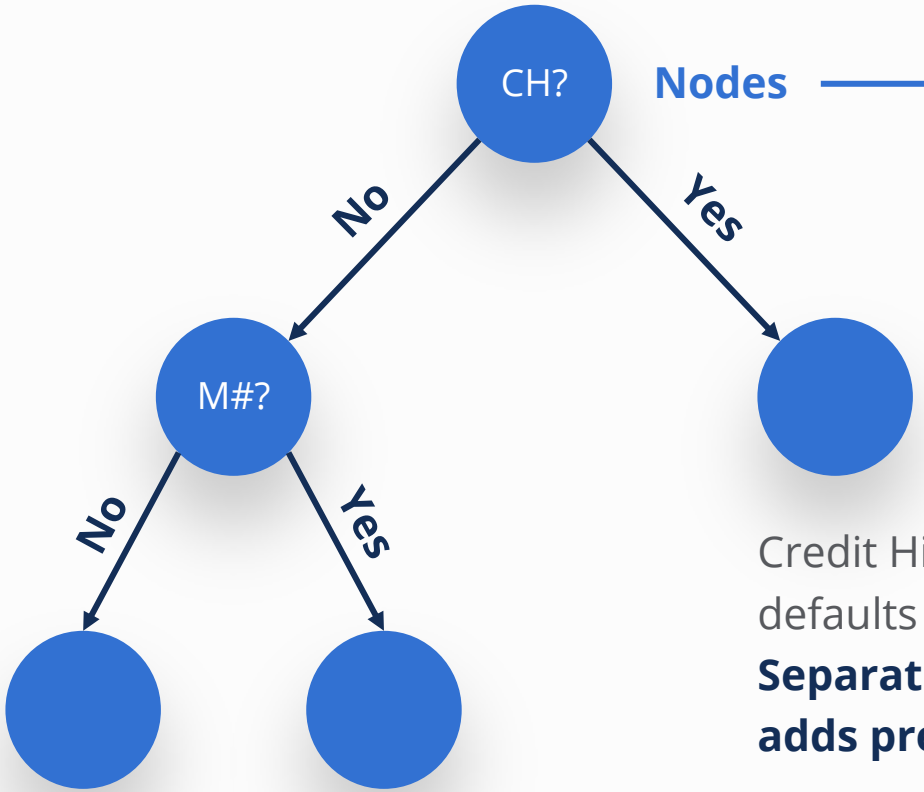
Identify overfitting in your results and learn how to adjust hyperparameters in your models to improve performance.

Ensemble model: model where results are based on the combined output of multiple other models to [hopefully] achieve a more reliable prediction.

Decision Trees

A decision tree is a machine learning technique that uses data to predict an outcome or classification.

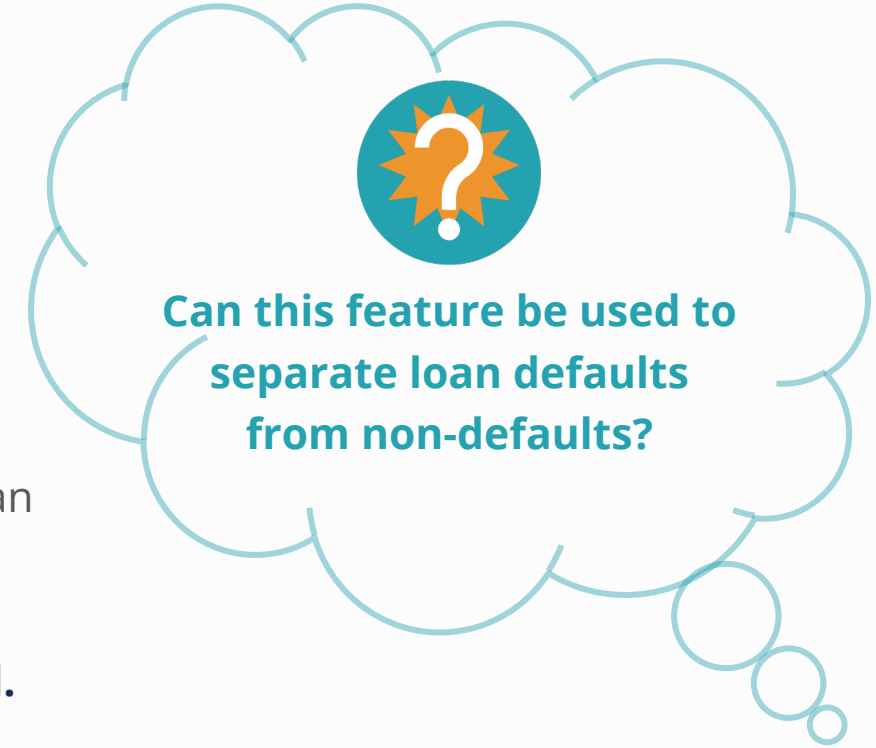
In this course, we are using information on car loans to predict which ones will default.



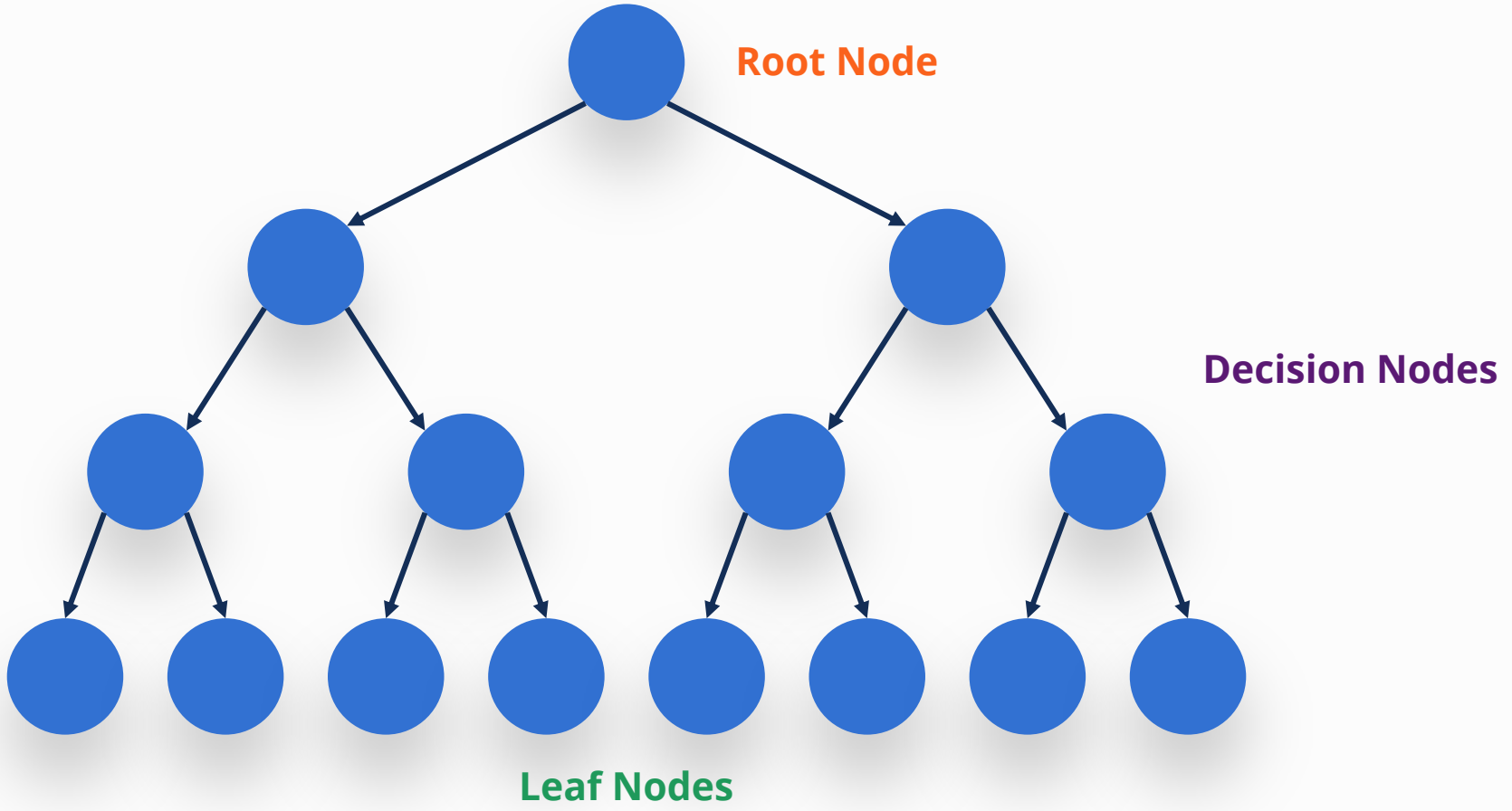
Improves separation of loan outcomes

Nodes → **Column**

Credit History (CH) helps separate loan defaults from non-defaults.
Separating loans according to CH adds predictive value to the model.



Decision Trees



Once the tree is completed, it can be used to evaluate each loan by answering the questions until a leaf node is reached and a prediction is made.

Decision Trees

A decision tree is a cascading set of questions, used to incrementally separate outcomes and improve predictive power.

How do you measure the performance at each node?

What happens when you have more than two outcomes?

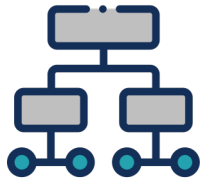
Do you have to use all the features each time?

Decision trees are relatively easy to visualize and interpret.

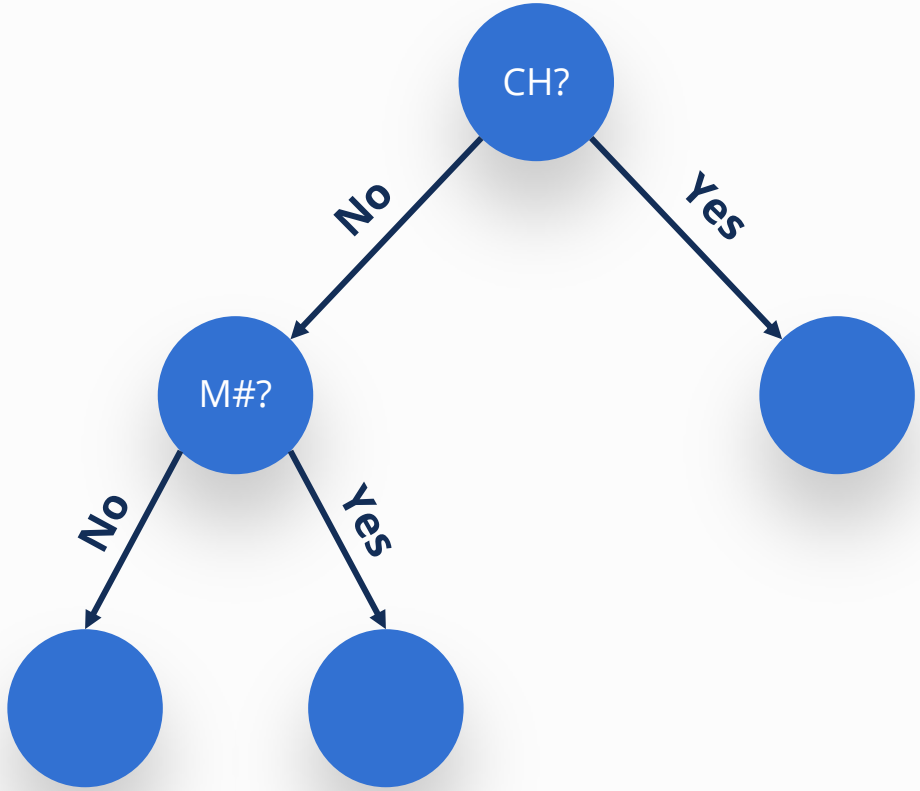
But they often have lower prediction accuracy compared to other models.

Random Forest

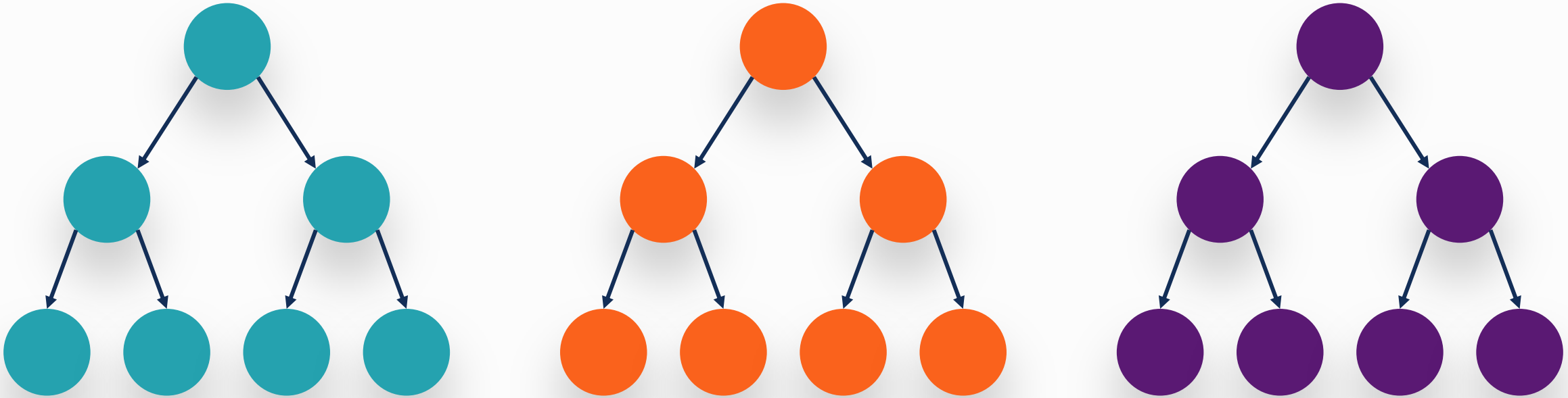
A random forest is known as an ensemble model since it combines the results from multiple other models; in this case decision trees.



A decision tree is a cascading set of questions, used to incrementally improve the classification of outcomes.



Random Forest



Predict 1 | Predict 0 | Predict 1

Predict 1

Random Forest

The random forest model can be fine tuned by changing **the number of trees we want to include in the forest, or the size of each tree. These criteria are known as hyper-parameters.**

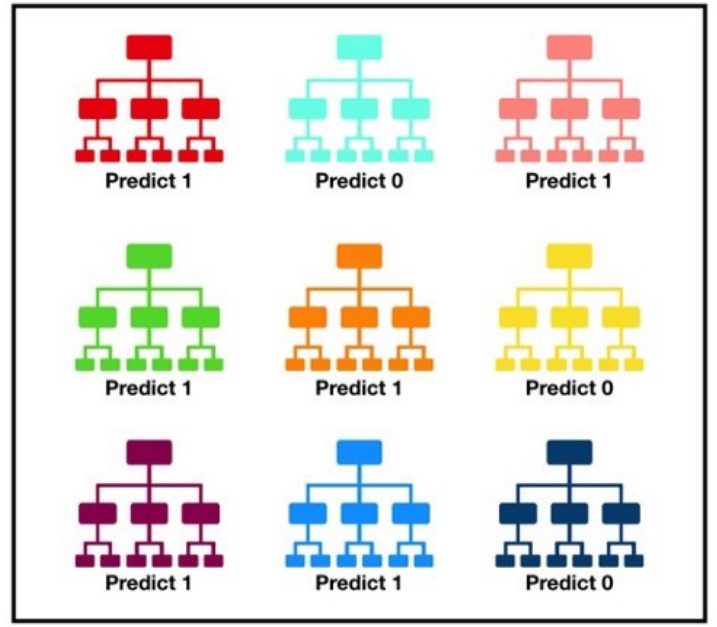
A random forest classifier is a machine learning technique used for predictive analysis.

It's predictions are based on the predictions of a number of underlying decision tree models.

This approach of combining multiple models often improves accuracy, but makes them more difficult to interpret.

Random Forest

- Random Forest is an ensemble learner that generates predictions using a group o
- To predict the class of some input data, each tree produces a prediction and the R popular class as its output
- Individual trees in the random forest should be uncorrelated, this means that they
  - The output of some trees may be wrong but others will be right
  - As a group the forest can produce better results than individual trees



Random forest performs well when

- There is some recognizable pattern within our feature set that distinguishes one class from the other
- Errors made individual trees have low correlations with each other

Building the Forest

- How does Random Forest ensure low correlations between individual trees?
  - Using a process known as Bagging or Bootstrapping

What is Bagging?

- Unique training sets are generated for each tree by randomly sampling the original training set with replacement
- This exploits the tendency of decision trees to overfit on their training data
- Each tree is trained on a unique training set, create diversity among the forest

Feature Randomness

- Unlike regular decision trees, trees in the random forest choose which feature to split on from a random subset of all the training features
- Again forcing diversity between individual trees

Bias and Variance

- There are some key machine learning concepts we have not discussed yet

Bias

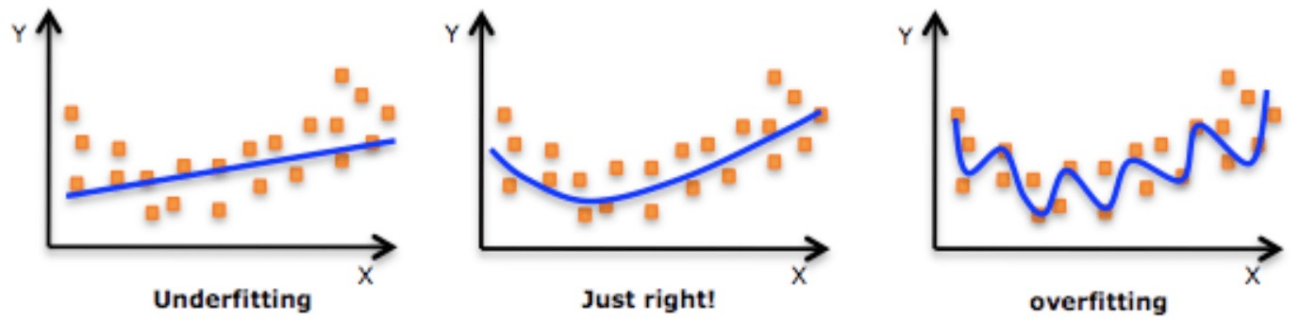
- How far are the predicted values from the actual values
- High bias usually means the model is too simple and misses important relationships between input features and the target variable

Variance

- The variability of predicted values if the model is shown different training data
- High variance means the model performs well on training data but does not generalize out to test/unseen data

Underfitting and Overfitting

- Overfitting and Underfitting are two important ideas to consider when investigating the performance of a machine learning model



Underfitting

- Model can not recognize patterns and associations in the training data
- Typically caused by a lack of complexity (e.g. low number of input features)
- Underfitted models have high bias and low variance
- Common when attempting to fit linear models to non-linear data

Overfitting

- Model is too complex and mistakes noise in the training data for common patterns and associations
- The model performs well on training data but does not generalize to test/unseen data
- Model is a complex representation of a simple reality
- Low bias and high variance

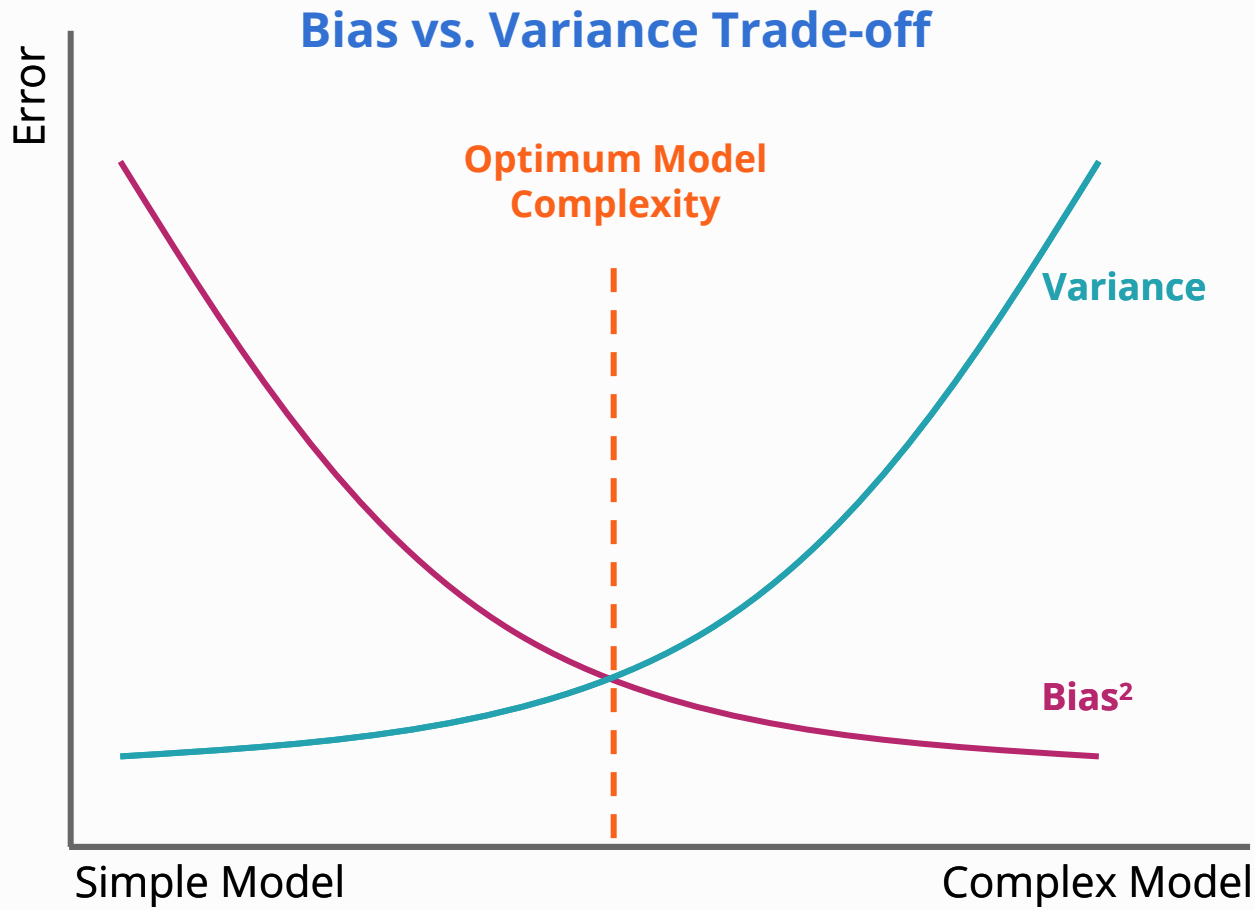
Bias vs. Variance

Bias is the difference between the prediction and the actual value.

High bias can be reduced from regularization.

Variance is the difference in fits in between datasets.

High variance can be resolved by reducing complexity.



High Bias
Low Variance
(Underfitting)

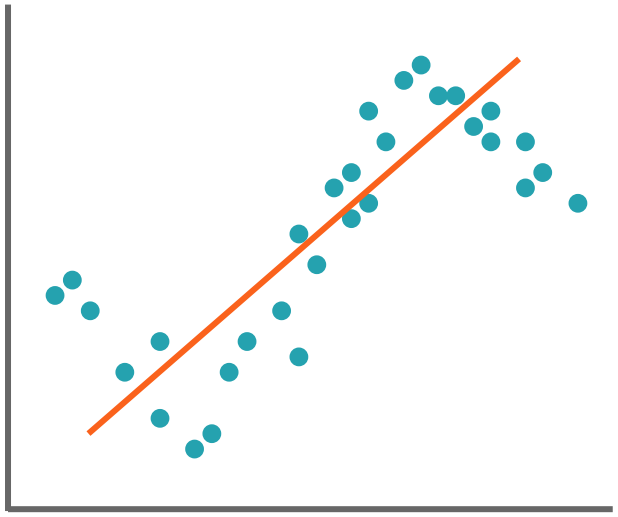
High Variance
Low Bias
(Overfitting)

Underfitting vs. Overfitting

The purpose of machine learning is to find the **real relationship** between inputs and outputs.

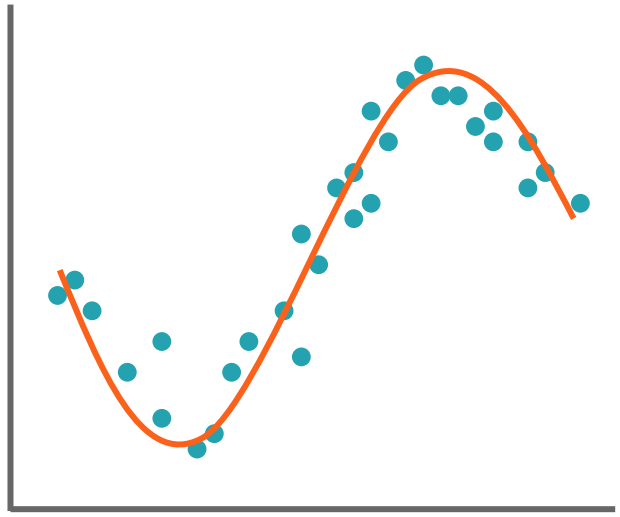
Underfit

Over generalizes the data



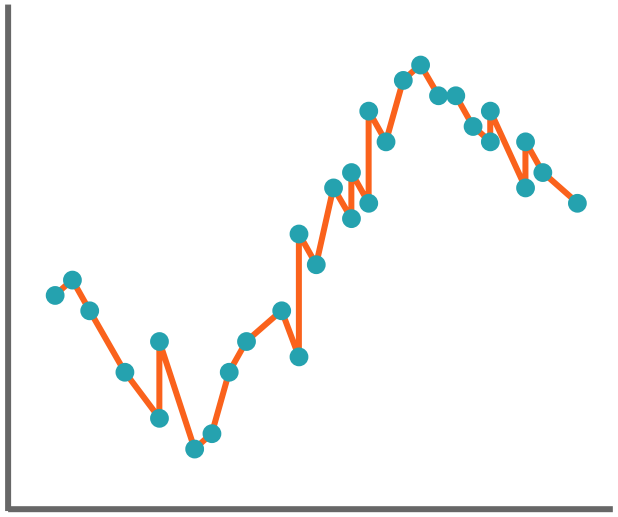
Good Fit

Generalize enough to predict future patterns



Overfit

Does not generalize enough



Ensemble Learning

- Ensemble learning is the idea that we can achieve more reliable predictions by combining the output from multiple machine learning models
- By combining multiple models together ensemble methods can minimize bias and variance

Ensemble Techniques

- Imagine we have a group of models are outputting their own classification results
- How do we decide on the final output?
  - We can simply take the mean, mode or weighted average
  - More complex techniques such as Bagging or Boosting can also be used

Pros and Cons

- Ensemble learners can produce more accurate results from stable robust models
- However, the resulting models often have high computational cost and can be difficult to interpret

Chapter Summary

Random forest classification model is an alternative way to classify loans.

Random Forest models tend to perform best when we pay attention to hyperparmeters such as:

Size of the forest

Depth of the trees

Created functions to help speed up the model build and evaluation process

Random forest classification model is an important and popular ML tool that is widely used in data science.

Next up...  **Improving Accuracy**

Improving Accuracy

Chapter Introduction

There are far fewer defaulted loans vs. non-defaulted loans.

Imbalance Classification Problem

- We're interested in the defaulted loans, however have the least amount of data on them.
- We need a fairer distribution of classes to help make predictions.

This is a common problem which applies to loan defaults, fraud detection, disease detection, among other areas.

Chapter Introduction

Learn how to use class balancing in the random forest & logistic regression models, as well as how to interpret the results.

We want to change the weighting of the model to focus more on the defaulting loans.

Motivation

- The distribution of loan defaults within our dataset is not even
  - 21.7% Defaulted Loans (Minority Class)
  - 78.3% Non-Defaults (Majority Class)
- In other words we have an imbalanced classification problem

Why Does it matter?

- In most imbalanced classification problems we are mainly interested in predicting the minority class
- This poses a problem, we want our model to pick up on the characteristics and patterns of the minority class but there are fewer instances to learn from
- Many real-world classification problems are in fact examples of imbalanced classification
  - Fraud Detection
  - Disease Detection
  - Loan Default Detection

Weight Balancing

- One method of dealing with imbalanced class labels is weight balancing
- Weight balancing negates the effect of imbalanced data by changing the weight that each class carries when computing the loss/error
- To balance the classes we can assign weights that are inverse to the class distribution
  - For our loan data we could assign a weight of 0.217 (21.7%) to class 0
  - And a weight of 0.783 (78.3%) to class 1

Up and Down Sampling

- A popular method of dealing with imbalanced class data is to resample the training data so that there is an equal number of instances of each class
- When using resampling techniques it is essential that the test data is not resampled.
  - We always need to test our model on the 'natural' class distribution

Up-Sampling

- Up-sampling means that we randomly duplicate instances of the minority class to create a balanced data set
- Most commonly we select a sample to duplicate without removing the original sample from the data set
  - This is known as resampling with replacement

Down-Sampling

- Down-sampling creates a balanced data set by randomly removing instances from the minority class without replacement

Synthetic Minority Oversampling Technique (SMOTE)

- Essentially SMOTE is a more sophisticated up-sampling technique
- Standard up-sampling creates duplicate instances which do not add any new information to the model

SMOTE

- Up-sample the minority class by creating synthetic samples with their own features
- Select samples that are close in the n-dimensional feature space
- Draw a line between similar samples and create a new synthetic sample at a point along the line

Congratulations!

Course Summary

Covered the basics of the entire machine learning process

Imported & cleaned data before conducting exploratory data analysis

Written code to **create a simple logistic regression & random forest classification model**

Explored techniques to evaluate and enhance the performance of your model

No machine learning model is perfect! It has to be fine tuned to maximize/minimize the errors that are most important to us.