# MIS710 – Machine Learning in Business
# Topic 4: Supervised Machine Learning **Logistic Regression**

Associate Professor Lemai Nguyen

AACSB ACCREDITED

DEAKIN BUSINESS SCHOOL

# Assignment 1 updates

- Check the unit site for REGULAR updates
  - A1 new due date: 12th August 8.00pm AEST, 3:30pm IST
  - Consultation sessions in Week 4 and Week 5

- Fundamental knowledge and skills - Lecture 3 and Lab 3 are critical to complete A1
- Industry expert joins us in Week 4 class
- Check A1 rubric

- Assignment extension requests are processed centrally, beyond the UNIT TEAM
- TurnItIn, submit your own work and cite sources properly
- Report writing improvement: Feedbackfruits

**Week 3:**

•Python basics: **26th July 3.00-4.00 pm AEST, 10.30-11.30 am IST,** room B4.01 and ZOOM: Dat Le.

**Week 4:**

•A1 consultation session 1: **Tuesday 30th July 3.00-4.00 pm AEST,** room HE1.008: Dat Le

•A1 consultation session 2: **Wednesday 31st July 7.30-8.30 pm AEST, 3.00-4.00 pm IST**, ZOOM: Emran Ali.

•A1 consultation session 3: **Friday 2nd August 3.00-4.00 pm AEST**, room B4.01: Thuc Nguyen

**Week 5:**

•A1 consultation session 4: **Monday 5th August 2.00-3.00 pm AEST**, room HE1.009: Durgesh Samariya

•A1 consultation session 5: **Tuesday 6th August 3.00-4.00 pm AEST**, room HE1.008: Dat Le

•A1 consultation session 6: **Tuesday 6th August 5.00-6.00 pm AEST, 12.30-1.30 pm IST**, ZOOM: Abhishek Jha

•A1 consultation session 7: **Wednesday 7th August 7.30-8.30 pm AEST, 3.00-4.00 pm IST,** ZOOM : Emran Ali.

All Zoom sessions are open to ALL - Online, GIFT city, and Burwood students. Zoom links can be found under Content/Online Classroom and Recordings.

AICafe, MLCafe, and A1 Forum are open to ALL - Online, GIFT city, and Burwood students.

# Preetham D Bangera

## Product Manager, Jio Platforms Limited


Preetham D Bangera is a Product Manager with extensive experience working in Digital Platforms, including his current role working with a cloud gaming platform at Jio, poised to revolutionize the gaming experience in India.

He has shown his expertise in Digital Marketing, Marketing Research and Technology Development over his roles at Boeing, PricewaterhouseCoopers (PwC) and now Jio.

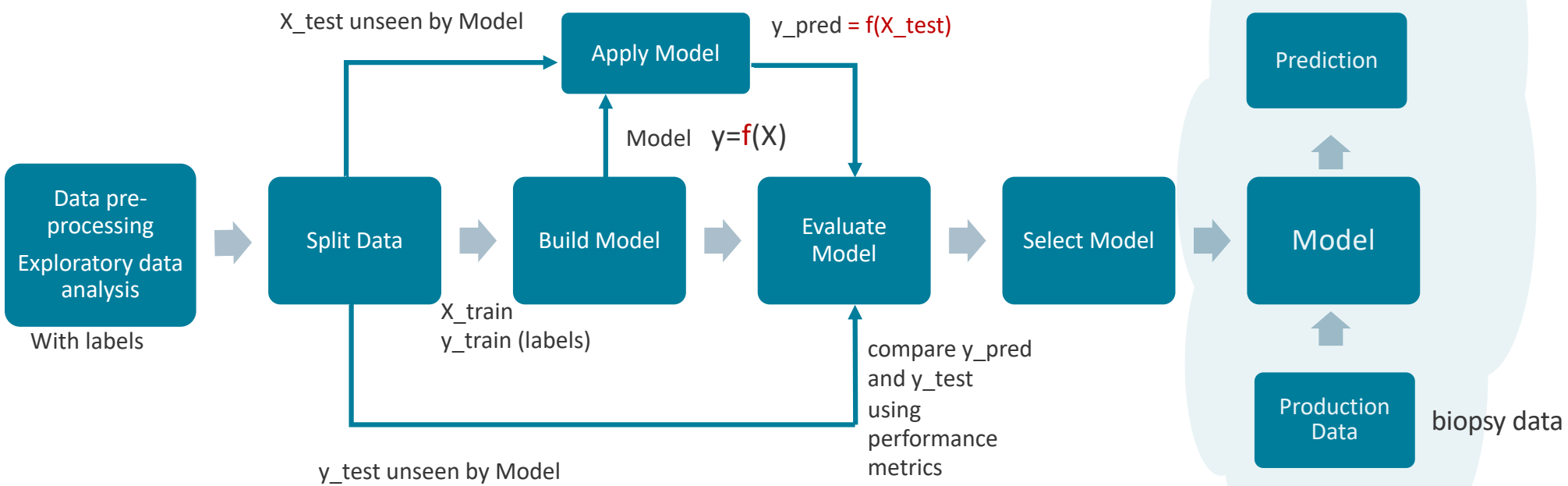
He holds a Master of Business Administration from Indian Institute of Management Indore.

https://www.linkedin.com/in/preetham-d-bangera-953a23110/

# Assignment 1 Part 1: EDA

1. *General information and game configuration*: Summarise the kind of games in the dataset, in terms of game types, year of release, age category, minimum number of players required, and maximum number of players allowed.
2. *Game engagement*: Summarise the average play time? Are there outliers?
3. *Game engagement and rating*: Is there a relationship between the playing time and the average ratings?
4. *Game complexity and rating:* Is there a relationship between level of game complexity and the average ratings?
5. *How do game configuration, popularity and Interest* (e.g., minimum number of players required, maximum number of players allowed, and number of owners, number of traders, numbers of interests and high interests) correlate with *rating*? For example, are games purchased by more users likely to result in higher ratings?
6. Additional insights regarding data quality, other variables and relationships.

# Assignment 1 Part 1: ML




# Assignment 1 Part 1: Report

- A cover page (**not** included in the word count) that includes:
  - Report Title, Unit code and name, Student name and student ID
- A table of contents (**not** included in the word count)
- An executive summary of **approx**. 200 words is required (**included** in the word count).
- The report should include:
  1. Business understandings including the business problem to address and other Business Analysis Core Concept Model (BACCM) elements in relation **the case study**.
  2. Data understanding, data preparation, exploration, visualization, and insights gained – including the Client's questions.
  3. The machine learning approach undertaken.
  4. The model and performance metrics.
  5. Discussion of the pros and cons of the model.
  6. Business solution and recommendations for implementation and improvement (based on the model).
- References using the APA7 style (**not** included in the word count)
- Optional appendices (**not** included in the word count – not subject to assessment)

# Executive summary

- **Who** commissioned the project (the Client) and **who** delivered (you!)
- State the **problem** to be addressed
- State the chosen **approach**
- Summarise **findings - solution**
- So what: Brief and specific **what the Client should do** and **Value** to be gained

## Executive Summary

This report presents the findings of the research project aimed to gain an understanding of the current maturity state of data analytics and its perceived business value in Australian organisations.

The research project was conducted by a research team at Deakin University, Deakin Business School, and in collaboration with Tata Consultancy Services (TCS) as the knowledge partner. The project was headed by Associate Professors Lemai Nguyen, Ambika Zutshi and William Yeoh (Deakin Business School) and embedded in Tata Consultancy Services (TCS) Global Research & Development program.

A cross-sector online questionnaire was conducted with 138 managers at various levels from the executive to senior and middle managers in data and analytics. Eight semi-interviews and a workshop to discuss findings were also undertaken. All the interview and workshop participants were executive-level and senior Information and Technology leaders.

### Key Findings

- An optimistic landscape of moderately medium and high levels of data analytics maturity in Australian organisations was self-reported by participants.
- Data and analytics were reported to deliver business value and enable business performance.
- The COVID-19 pandemic reinforced the role of data and analytics for business survivability and performance.
- A value-driven maturity journey, inclusive data and analytics culture and a balanced data governance approach were found to be key elements in leveraging data and analytics for business gains.

The findings presented within this report shed light on the current state of data and analytics maturity across various sectors in Australia. We hope that it will inspire and offer insights to guide business decision-makers on the value, benefits, and best-practices of adopting data and analytics on their digital transformation journey.

## Research Approach

A mixed-methods research approach was adopted. It consisted of:

- Online questionnaire survey completed by 138 participants representing Chief Information or Technology Officers (CIOs / CTOs), other executive positions, and senior and middle managers.
- Eight (8) semi-structured interviews conducted with Executive managers, Directors, Heads, and senior managers.
- A workshop conducted with twelve (12) participants to discuss the preliminary survey findings.

All responses were self-reported by and reflect the participants' professional views[1]. The project was executed in accordance with the Deakin University Ethics Committee approval. Only de-identified and collective findings are presented in the report except where permission has been granted to disclose the participant's details.

*Nguyen et al, 2022*

1. Demographics information of the participants can be found at the Section—Demographics Summary.

4

# Executive summary

- Brief and specific what the Client should do and Value to be gained
- How to improve the model in the future

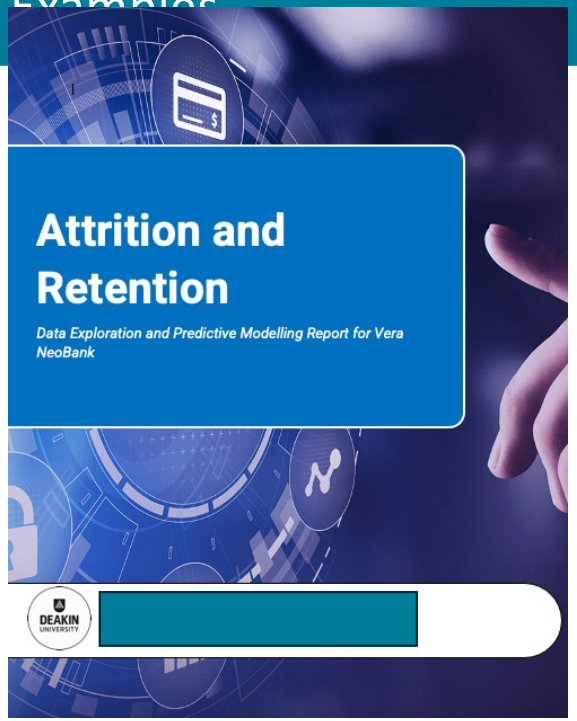
Based on our findings and the given model, we recommend the following process changes for implementation:

- **Customer Service Interdiction:** Customers flagged predictively for 'at-risk' should receive additional outreach by representatives.
- **Reducing Overdraft Penalties:** Enabling increased tolerance for overdraft by customers can improve retention, as 'Active' customers have higher outstanding balance and may remain due to obligations….
- **Self-Service Facilities:** Number of contacts was present in Cluster 0 with lower activity. Instead of increasing headcount for customer services, the company may elect to implement more self-service resources to avoid reduction of activity and reduce usage frictions.

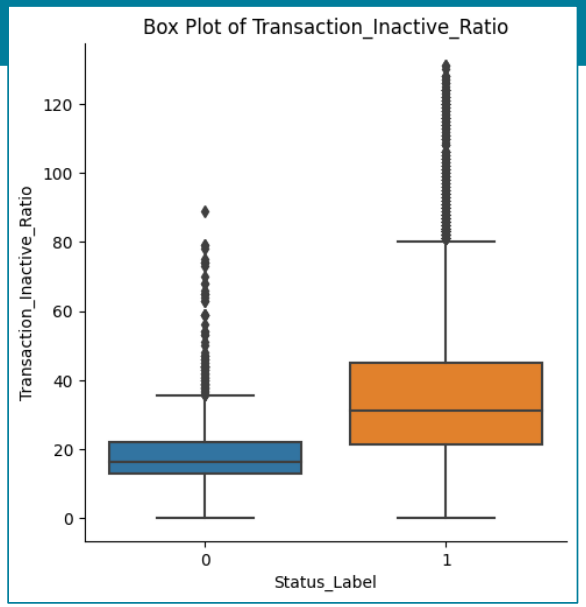
For further improvements, we recommend to collect and transfer time-series data, expanded scope of attributes, and increased sample sizes to improve findings in the future.

**(Adapted from Emre Deniz MIS710, 2023, Report B)**

# Examples

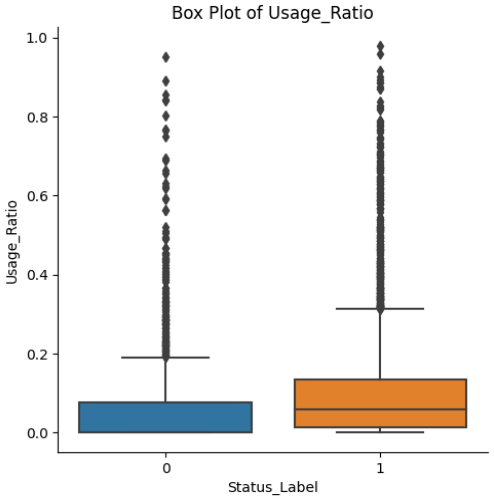



**(Extracted and adapted from Emre Deniz MIS710, 2023, Report A)**




… We find that closed accounts have higher inactivity and higher number of con
months.

Further, we find th
of accounts in clo
distinctly lower.




**Usage Ratio:** is more complex version of the Utilization Ratio. We computed this as the average transaction amount (Amount/ Frequency of Transactions) divided by the number of opened accounts and multiplied by the utilization. Min, Q2 and A3 are

***Model recommendation***

*Overall, the performance metrics indicates that the Model XYZ is good at determining class discriminants between Status, and this is largely consistent with the research literature [citations..] where the features and sample sizes are restricted and similar….*

# Assignment 1 Part 3
# Competition - Build your portfolio!

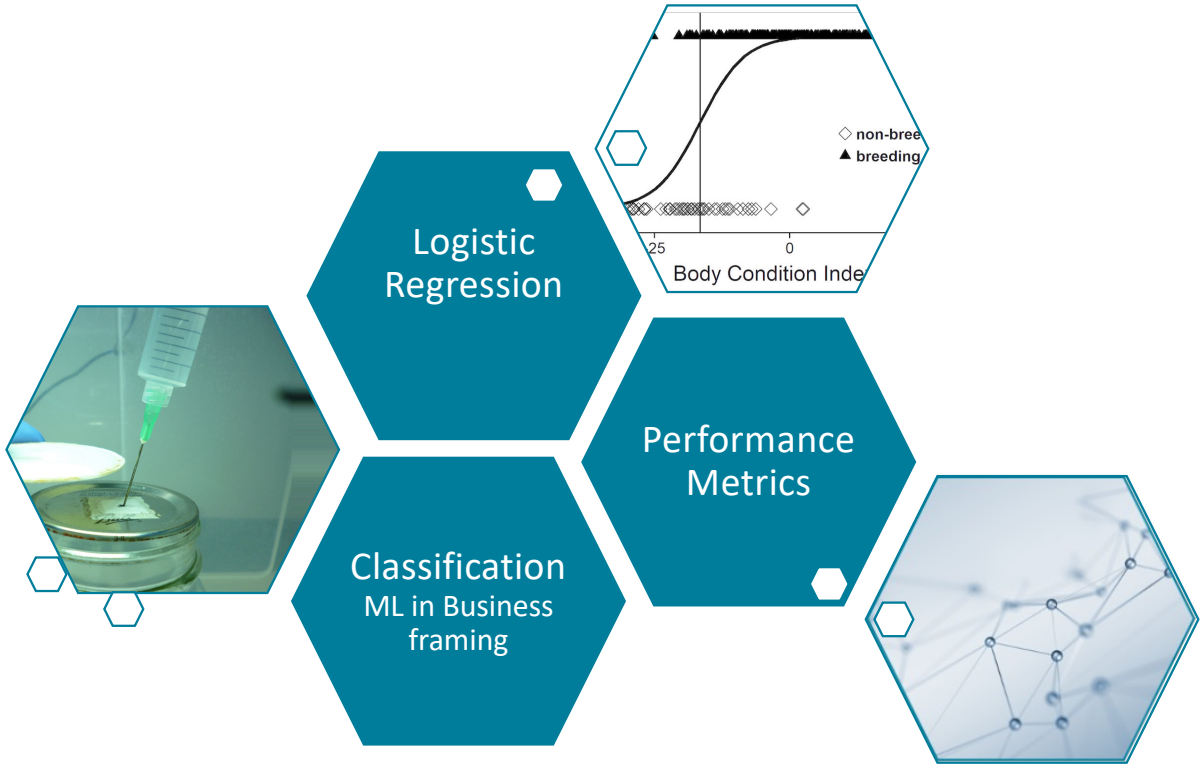

**Optional** Part 3: If you participate in the A1 Challenge then submit a csv deployment file with your predicted labels.

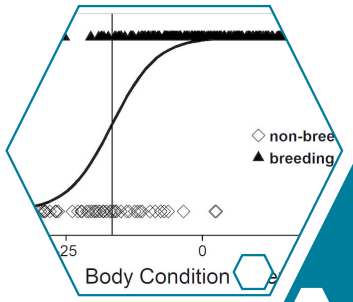
# See how you're pacing!


https://www.mentimeter.com

**Code: 6579 9913**

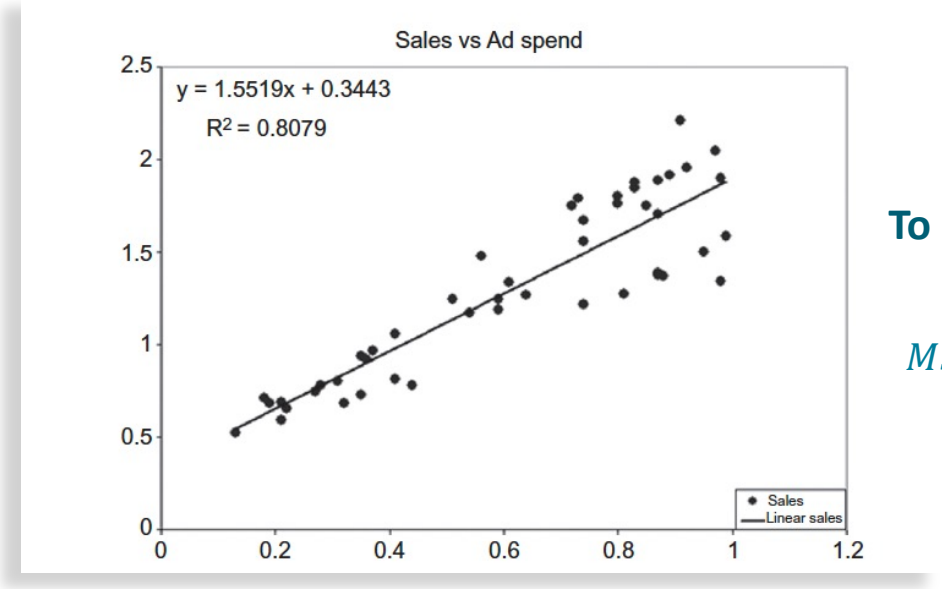
# Predictive Machine Learning with Logistic Regression






Logistic Regression

# RECAP: Training a simple linear regression model



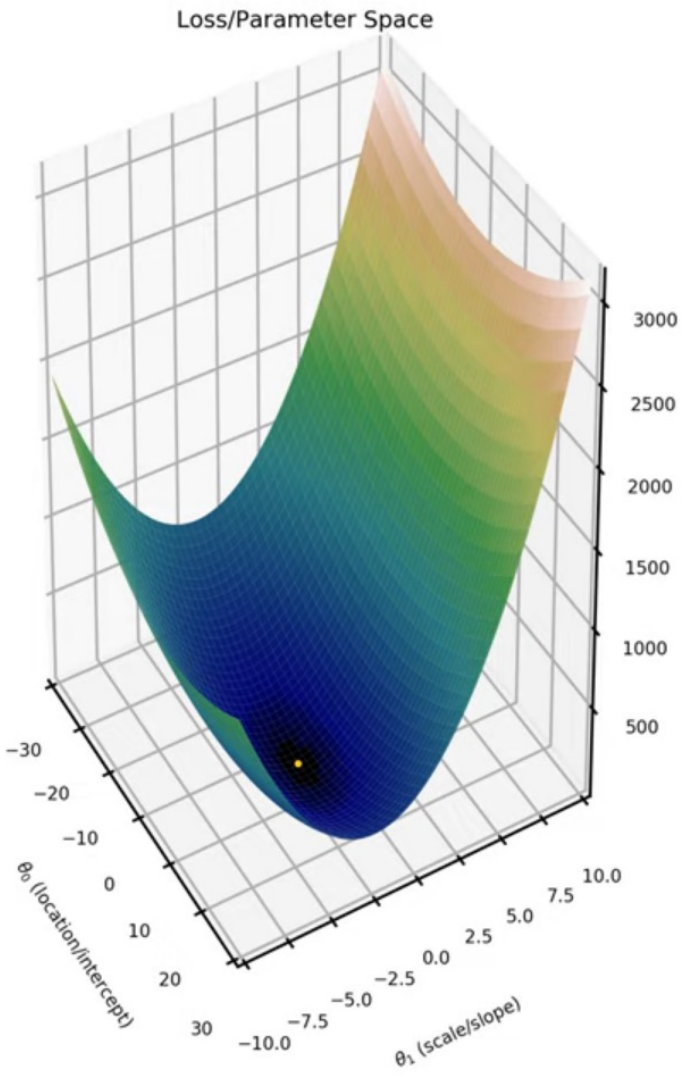

*Kotu and Deshpande, 2019*

Continuous target data

$$\hat{y} = b_0 + b_1 x$$

**To minimise the errors**

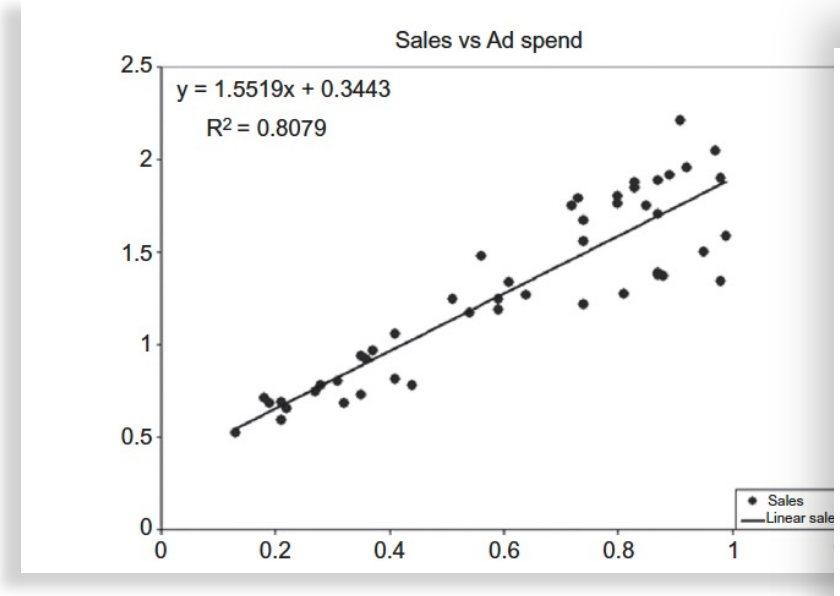
$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$$




https://www.youtube.com/watch?v=-JkE8U62aN0

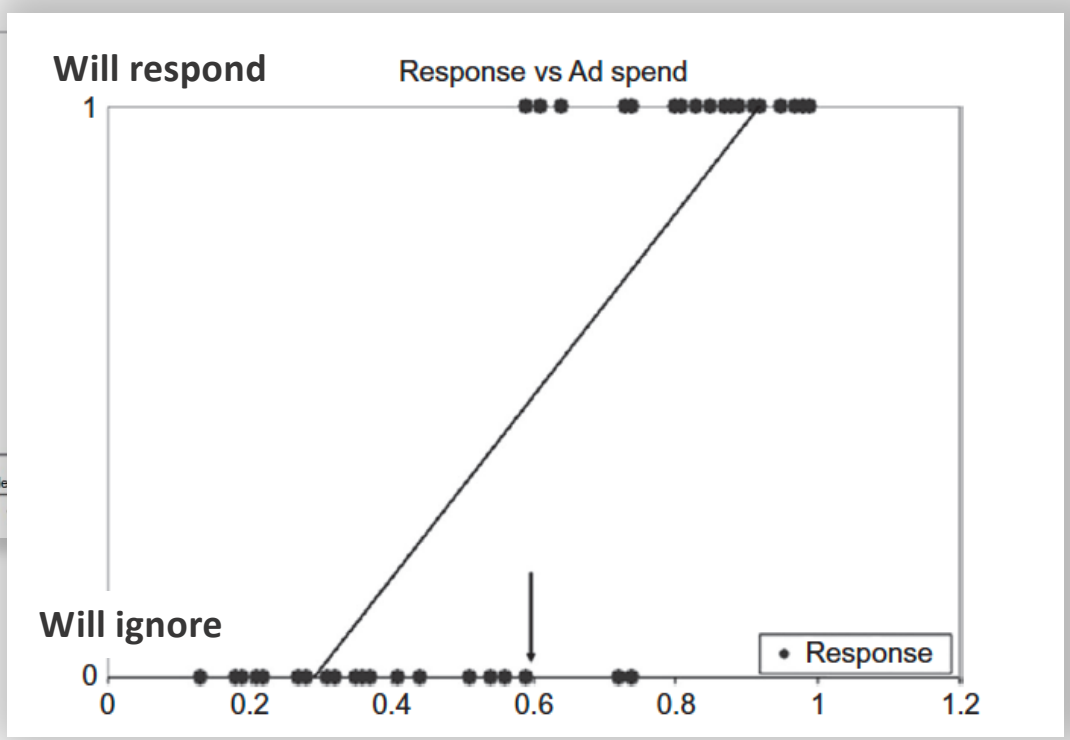
# Estimation vs Classification

*Kotu and Deshpande, 2019*




Continuous target data

$$\hat{y} = b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_n x_n$$




Discrete target data

# Fitting Linear Regression to a binary target

*Kotu and Deshpande, 2019*

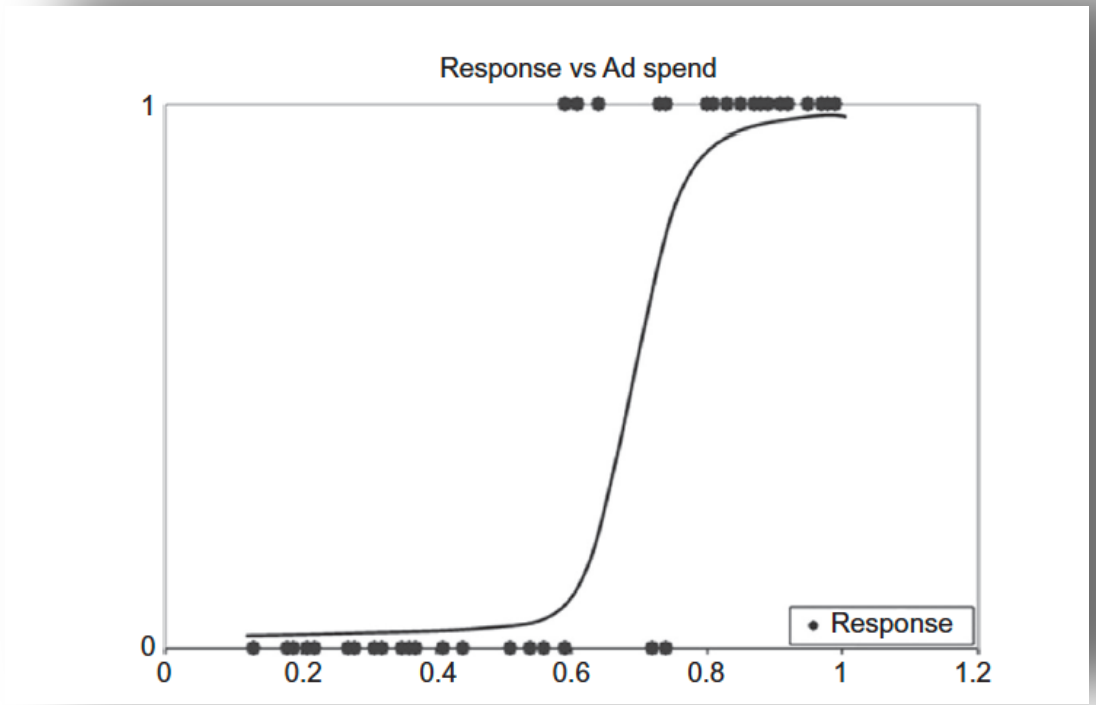
- **Target is categorical**
- Predictors can be continuous or categorical

One way is to convert

$$b_0 + b_1x_1 + b_2x_2 + \cdots + b_nx_n$$

into $p \in \{0,1\}$

$\hat{y}$ { 1 if p> 0.5 (**or another threshold**)
0 otherwise



# Odds, Logit, and Sigmoid function

- p is the probability of the event y happening
- (1-p) is the probability of the event not happening

p/(1-p) -> the odds (**odds ratio**) of the event happening

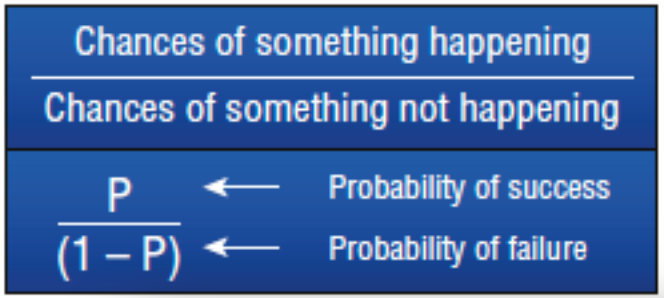
**log of the odds** log (p/(1-p)) is called is called the **logit**

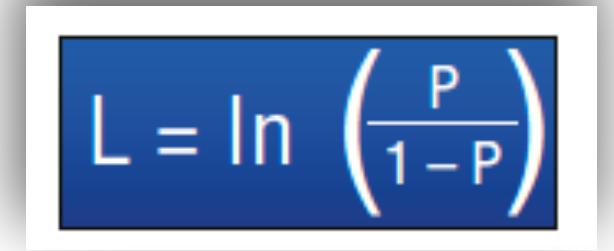
$$\text{logit} = b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_n x_n$$

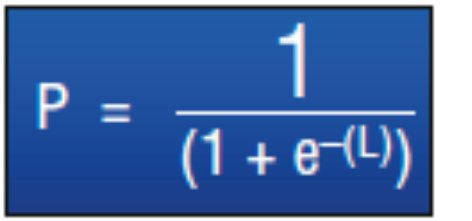
Logit is continuous from - ∞ to + ∞

$\boldsymbol{p}$ can be calculated using the **Sigmoid function**:

$$p = e^{\text{logit}} / (1 + e^{\text{logit}}) \qquad = 1 / (1 + e^{-logit})$$



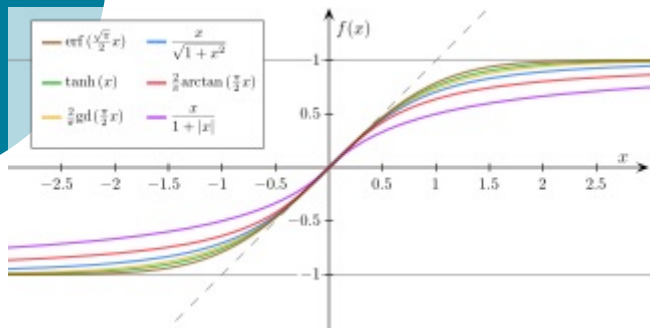




Lee (2019)

e= 2.718281828459

$log_b(b^x) = x$

# The Sigmoid functions

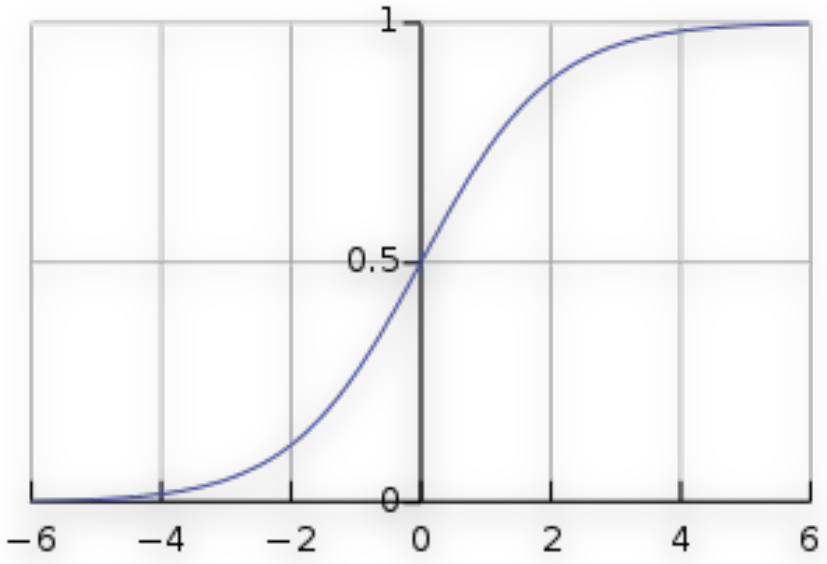



A common Sigmoid function is the logistic function

$$S(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{e^x+1} = 1 - S(-x)$$

**x** is continuous from - ∞ to + ∞

If x= - ∞ then Sigmoid (x) = 0
If x= + ∞ then Sigmoid (x) = 1




https://en.wikipedia.org/wiki/Sigmoid_function

❖ Aha!!
**The Sigmoid can convert the following into a number between 0 and 1**

$$b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_n x_n$$

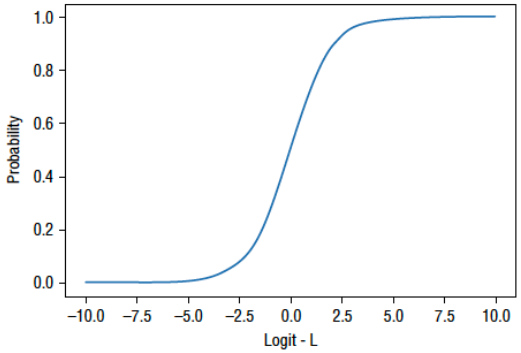
# Logistic Regression
## How it works

$$P = \frac{1}{(1 + e^{-(L)})}$$

Given predictors X, logit is a linear regression

$$\text{logit} = b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_n x_n$$

$$P = \frac{1}{(1 + e^{-(b_0 + b_1 x)})}$$

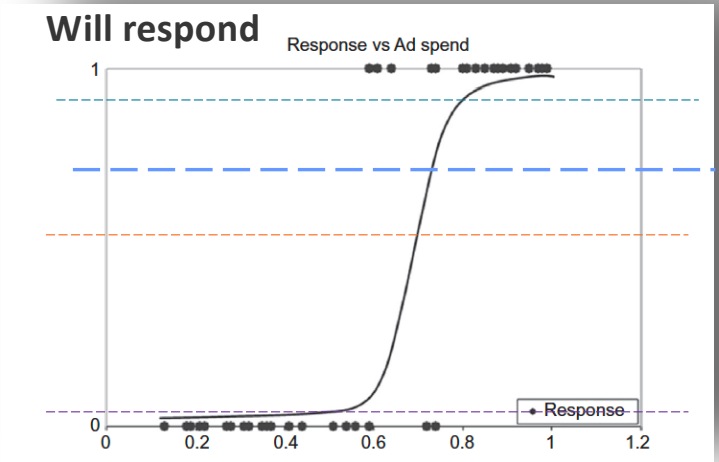



*Lee (2019)*

Probability ($\hat{y}$) can be computed using the **logistic function (Sigmoid)**:

$$p = e^{\text{logit}} / (1 + e^{\text{logit}}) = 1 / (1 + e^{-logit})$$

$$\hat{y} \begin{cases} 1 \text{ if } p > 0.5 \text{ (or another threshold)} \\ 0 \text{ otherwise} \end{cases}$$

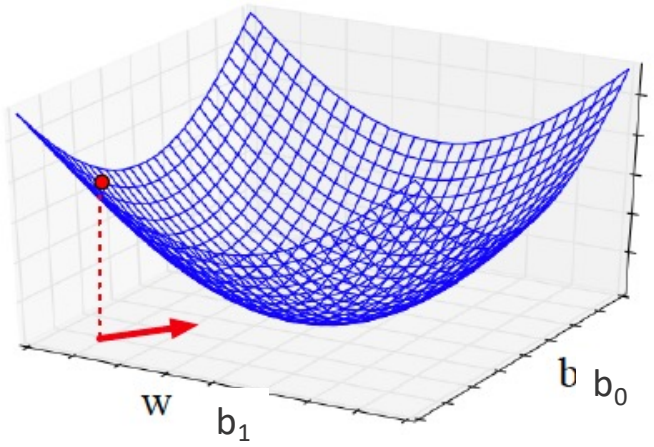



*Kotu and Deshpande, 2019*

# Training the logistic model

Model training and loss, cost functions

Training a logistic regression model involves searching for the coefficients $\beta_i$ that minimize the cost function - the average of the negative log-likelihood over all training examples.




Jurafsky & Martin (2023)

- Cost function measures how much $\hat{p}$ differs from the **true y**
- Gradient descent can be utilised to search for coefficients to maximise the likelihood of correct estimations

# Classification
## ML in Business framing

# Business understandings using BACCM


**Need:** *long delay in biopsy analysis process; shortage of experienced pathologists*

**Solution:** to **augment decision-making by** predicting whether biopsy samples are cancerous (malignant) or healthy

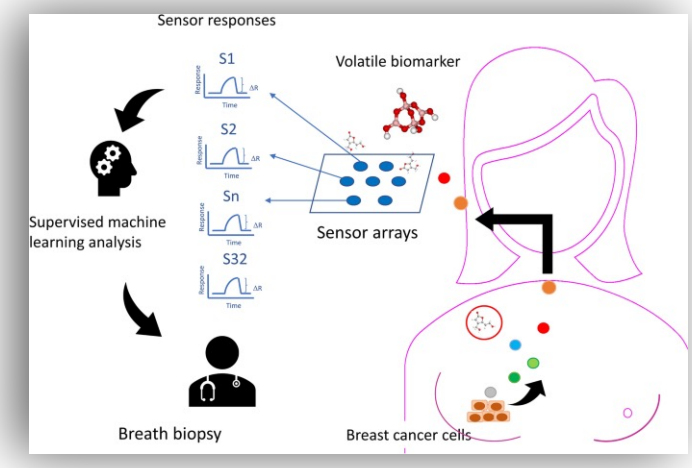
**Value:** *reduced analysis time, reduced delay, provide decision support for new pathologists*

**Change:** *augmentation of the biopsy data analysis process*

**Stakeholder:**

*Pathology labs, data analytics teams, pathologists, health practitioners, patients and other health consumers*

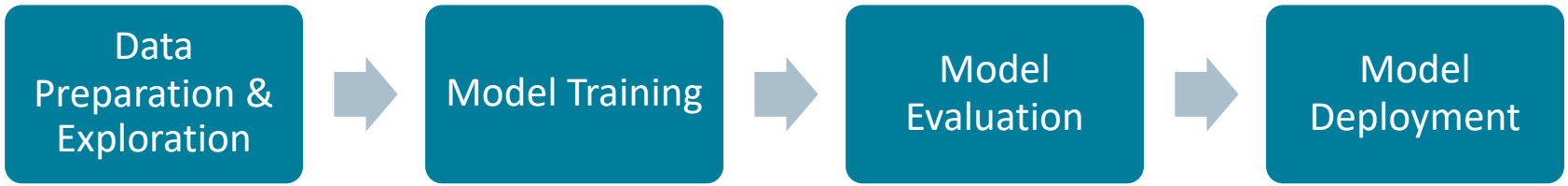
**Context:** pathology process, the Institute of Health and Nursing Australia *500 million pathology tests are conducted in Australia every year.*




https://www.nature.com/articles/s41598-020-80570-0

# ML Type and Problem Framing

- Business Context: Pathology Labs
- Business Problem: To reduce biopsy analysis time, with the same or higher level of accuracy
- Business Data: Historical datasets of biopsy data and diagnoses (label)
- Machine Learning type: Supervised
- Problem: Classification (which one – predict a categoric value)



# Dataset

**Data**:
V1, V2, V7, V8, V9: biological variables
Diagnosis: healthy or cancerous

Sample size: 699
Number of columns: 7
V1, V2, V7, V8, V9: integer
**Label:** diagnosis: string

**Source**: **adapted** from a dataset provided by Dr Mark Griffin, Industry Fellow, University of Queensland

dependent variable, label (y)

| ID | V1 | V2 | V7 | V8 | V9 | diagnosis |
|---|---|---|---|---|---|---|
| 1177399 | 8 | 3 | 1 | 6 | 2 | healthy |
| 1246562 | 10 | 2 | 1 | 1 | 2 | healthy |
| 1108370 | 9 | 5 | 2 | 1 | 5 | healthy |
| 1165926 | 9 | 6 | 2 | 9 | 10 | healthy |
| 1167439 | 2 | 3 | 2 | 5 | 1 | healthy |
| 1226012 | 4 | 1 | 2 | 1 | 1 | healthy |
| 601265 | 10 | 4 | 2 | 3 | 1 | healthy |
| 1295186 | 10 | 10 | 2 | 8 | 1 | healthy |
| 1343068 | 8 | 4 | 2 | 5 | 2 | healthy |
| 1065726 | 5 | 2 | 3 | 6 | 1 | healthy |
| 1102573 | 5 | 6 | 3 | 1 | 1 | healthy |
| 1106829 | 7 | 8 | 3 | 8 | 2 | healthy |
| 1108449 | 5 | 3 | 3 | 4 | 1 | healthy |
| 1111249 | 10 | 6 | 3 | 6 | 1 | healthy |
| 1112209 | 8 | 10 | 3 | 9 | 1 | healthy |
| 1116116 | 9 | 10 | 3 | 3 | 1 | healthy |
| 1116132 | 6 | 3 | 3 | 9 | 1 | healthy |
| 1126417 | 10 | 6 | 3 | 2 | 3 | healthy |
| 1166654 | 10 | 3 | 3 | 10 | 2 | healthy |
| 1169049 | 7 | 3 | 3 | 2 | 7 | healthy |
| 1171845 | 8 | 6 | 3 | 1 | 1 | healthy |

# ML in Business Framing

**Business needs**

- Long delay in returning pathology results
- Novice pathologists need training

**Data**

- Sources: Past biopsy data and results
  - Internal and external
- **Data content: Biomarkers and diagnoses**

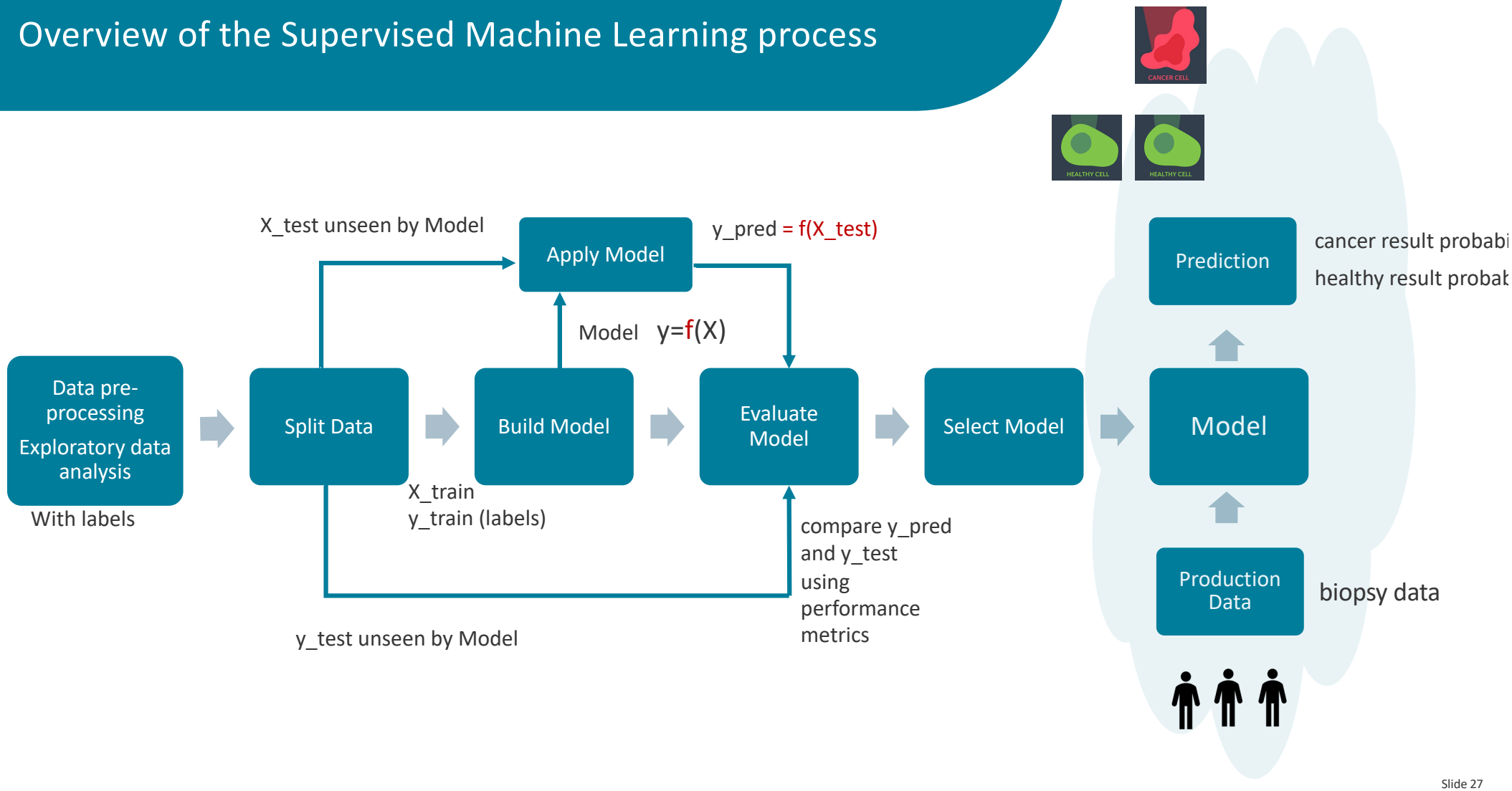
**Analytics (ML)**

- **Supervised ML, classification**
- Build and test a logistic regression model that best fit the historical biomarkers and diagnoses

**Deployment**

- Decision augmentation: in interpreting biopsy data -> reduced time and improved accuracy
- Training novice pathologists

# Overview of the Supervised Machine Learning process



# Loading and Exploring the Dataset

```
# load dataset
records = pd.read_csv("/content/drive/MyDrive/MIS710/biopsy_ln.csv")

records.info()

records.describe()
```

```
[11] records.info(10)

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 699 entries, 0 to 698
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   ID          699 non-null    int64
 1   V1          699 non-null    int64
 2   V2          699 non-null    int64
 3   V7          699 non-null    int64
 4   V8          699 non-null    int64
 5   V9          699 non-null    int64
 6   diagnosis   699 non-null    object
dtypes: int64(6), object(1)
memory usage: 38.4+ KB
```

1 to 8 of 8 entries Filter

| index | ID | V1 | V2 | V7 | V8 | V9 |
|---|---|---|---|---|---|---|
| count | 699.0 | 699.0 | 699.0 | 699.0 | 699.0 | 699.0 |
| mean | 1071704.0987124464 | 4.417739628040057 | 3.13447782546495 | 3.4377682403433476 | 2.866952789699571 | 1.5894134477825466 |
| std | 617095.7298192448 | 2.8157406585949327 | 3.0514591099541892 | 2.438364252324239 | 3.0536338936127385 | 1.7150779425067932 |
| min | 61634.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| 25% | 870688.5 | 2.0 | 1.0 | 2.0 | 1.0 | 1.0 |
| 50% | 1171710.0 | 4.0 | 1.0 | 3.0 | 1.0 | 1.0 |
| 75% | 1238298.0 | 6.0 | 5.0 | 5.0 | 4.0 | 1.0 |
| max | 13454352.0 | 10.0 | 10.0 | 10.0 | 10.0 | 10.0 |

Show 25 per page
Like what you see? Visit the data table notebook to learn more about interactive tables.

## Which variable(s) is irrelevant to be predictor?

1 to 8 of 8 entries Filter

| index | ID | V1 | V2 | V7 | V8 | V9 |
|---|---|---|---|---|---|---|
| count | 699.0 | 699.0 | 699.0 | 699.0 | 699.0 | 699.0 |
| mean | 1071704.0987124464 | 4.417739628040057 | 3.13447782546495 | 3.4377682403433476 | 2.866952789699571 | 1.5894134477825466 |
| std | 617095.7298192448 | 2.8157406585949327 | 3.0514591099541892 | 2.438364252324239 | 3.0536338936127385 | 1.7150779425067932 |
| min | 61634.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| 25% | 870688.5 | 2.0 | 1.0 | 2.0 | 1.0 | 1.0 |
| 50% | 1171710.0 | 4.0 | 1.0 | 3.0 | 1.0 | 1.0 |
| 75% | 1238298.0 | 6.0 | 5.0 | 5.0 | 4.0 | 1.0 |
| max | 13454352.0 | 10.0 | 10.0 | 10.0 | 10.0 | 10.0 |

Show 25 per page

Like what you see? Visit the data table notebook to learn more about interactive tables.

```
#drop the column ID
records=records.drop(['ID'], axis=1)
records.info()
```

# Data preparation

```
#convert categorical data to numerical
def coding_diagnosis(x):
   if x=='cancerous': return 1
   if x=='healthy': return 0

records['Diagnosis'] = records['class'].apply(coding_diagnosis)

print(records.sample(10))
```

| | V1 | V2 | V7 | V8 | V9 | diagnosis | Diagnosis |
|---|---|---|---|---|---|---|---|
| 343 | 3 | 1 | 1 | 1 | 1 | healthy | 0 |
| 617 | 1 | 1 | 3 | 1 | 1 | healthy | 0 |
| 153 | 10 | 10 | 7 | 10 | 10 | cancerous | 1 |
| 421 | 1 | 1 | 2 | 1 | 1 | healthy | 0 |
| 629 | 3 | 1 | 3 | 1 | 1 | healthy | 0 |
| 361 | 6 | 2 | 1 | 1 | 1 | healthy | 0 |
| 320 | 5 | 3 | 1 | 1 | 1 | healthy | 0 |
| 323 | 1 | 1 | 1 | 1 | 1 | healthy | 0 |
| 107 | 10 | 4 | 6 | 5 | 2 | cancerous | 1 |
| 354 | 5 | 1 | 1 | 1 | 1 | healthy | 0 |

# Data preparation

```
#Convert categorical variables to numeric
#Define the custom mapping
diagnosis_mapping = {
    'cancerous': 1,
    'healthy': 0,
}
# Convert the categories to numerical values using replace()
records['Diagnosis'] =
records['diagnosis'].replace(diagnosis_mapping)

print(records.sample(10))
```

| | V1 | V2 | V7 | V8 | V9 | diagnosis | Diagnosis |
|---|---|---|---|---|---|---|---|
| 343 | 3 | 1 | 1 | 1 | 1 | healthy | 0 |
| 617 | 1 | 1 | 3 | 1 | 1 | healthy | 0 |
| 153 | 10 | 10 | 7 | 10 | 10 | cancerous | 1 |
| 421 | 1 | 1 | 2 | 1 | 1 | healthy | 0 |
| 629 | 3 | 1 | 3 | 1 | 1 | healthy | 0 |
| 361 | 6 | 2 | 1 | 1 | 1 | healthy | 0 |
| 320 | 5 | 3 | 1 | 1 | 1 | healthy | 0 |
| 323 | 1 | 1 | 1 | 1 | 1 | healthy | 0 |
| 107 | 10 | 4 | 6 | 5 | 2 | cancerous | 1 |
| 354 | 5 | 1 | 1 | 1 | 1 | healthy | 0 |

# Recap: EDA

```
## **1.3 Conduct Exploratory Data Analysis EDA**

1. Univariate analysis

> Numeric variables

* Summarise data through mean, median (Q2), Q1, Q3, range, Interquartile Range (IQR),
outliers, standard deviation, and variance.

* Explore distributions using histograms, boxplots, kernel density estimates.


> Categorical variables

* Summarise data through mode(s), counts and percentages of unique categories.

* Explore bar charts, pie charts, or frequency tables of categorical variables.
```

# Recap: EDA

2. Bivariate analysis

3. Multivariate analysis

> Explore relationships between two or more variables

* Compare two continuous variables using their descriptive statistics (such as means, medians, and Inter Quartile Range (IQR), range) and explore the relationships between them using boxplots, scatterplots and correlation coefficients.

* Explore the relationships between two categorical variables using a two-way contingency table (crosstab), clustered bar charts, stacked bar charts or a mosaic plot.

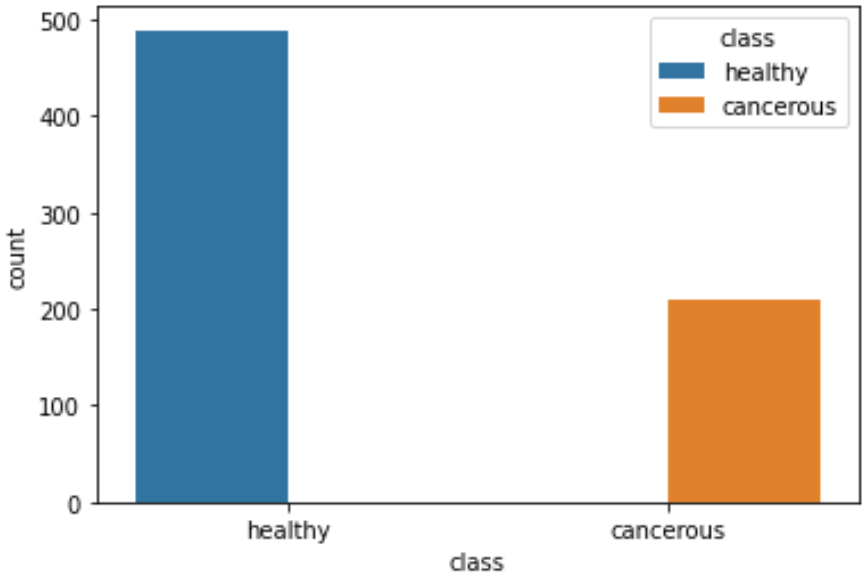
* Explore the relationship between one numeric and one categorical variable, through using and grouped boxplots, violin plots, and histograms.

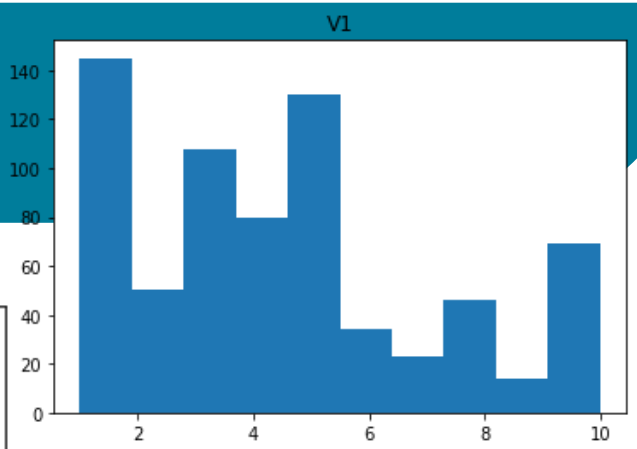
* Explore correlations among multiple (selected) numeric variables using heatmaps of the correlation matrix.

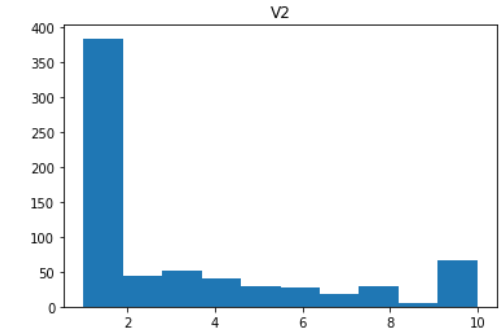
3. Convert data as needed

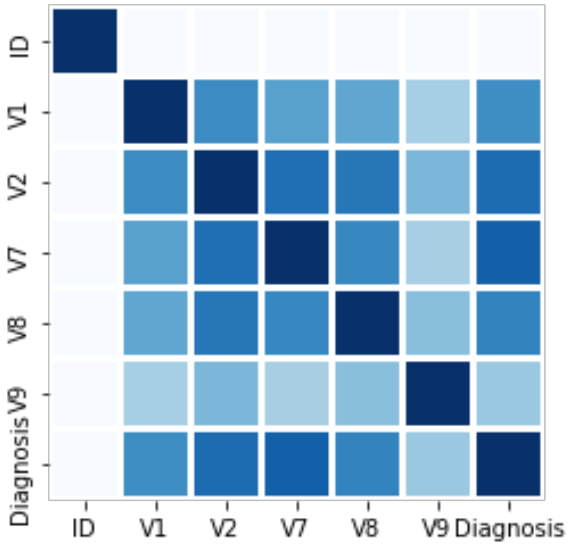
You can also explore logistic regression relationships between two variable

# Exploration



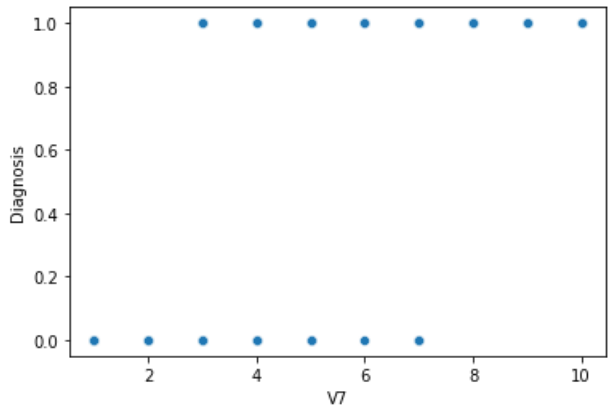



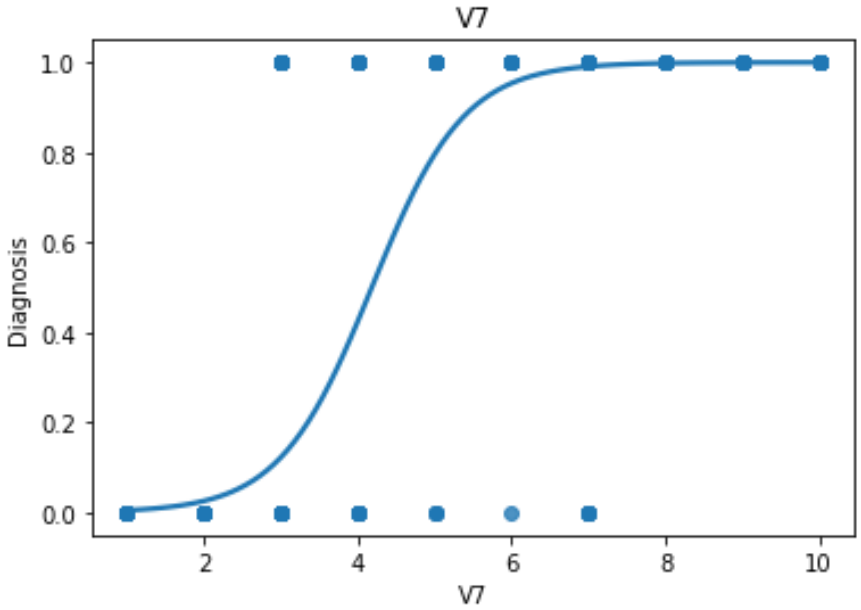

```
sns.countplot('class', data=records, hue='class')
```

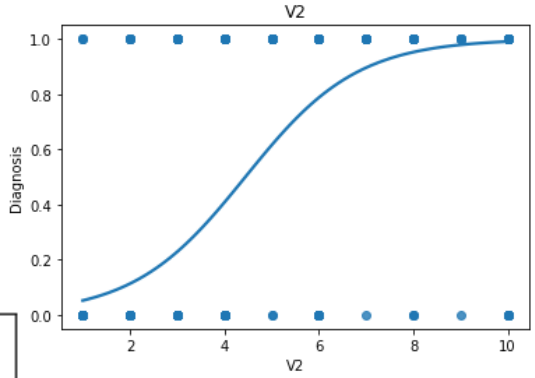
MIS770 and MIS771, see Lab 3 solution

# Exploration

```
for i in records.iloc[:,1:5]:
  sns.regplot(x=records[i], y=records['Diagnosis'],
  logistic=True, ci=None)
plt.title(i)
plt.show()
```



# Feature and label selection

```
#Selecting predictors
features = records.columns[0:5]
X=records[features] #predictors

#specify the label
y=records['Diagnosis']
```

# Data Splitting

```
# Import train_test_split function
from sklearn.model_selection import train_test_split

# Split dataset into a training dataset 80% and a test dataset 20%
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20,
random_state=2024, stratify=y)
```

**Consider**:

- Sample size
- Imbalanced classification

```
      V1  V2  V7  V8  V9
617    1   1   3   1   1
107   10   4   6   5   2
17    10   6   3   2   3
441    5   2   2   2   1
365    4   1   1   1   1
617    0
107    1
17     0
441    0
365    0
Name: Diagnosis, dtype: int64
Training dataset size: 559
Test dataset size: 140
```

# Model Training

```
from sklearn.linear_model import
LogisticRegression

#Create an initial Logistic Regression model
logreg = LogisticRegression(max_iter=100)

# Train Logistic Regression Classifier with the training dataset
logreg = logreg.fit(X_train, y_train)
```

https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

# Apply the model to make predictions

```
#Make predictions for the test dataset
y_pred = logreg.predict(X_test)
```

```
#join unseen y_test with predicted value
inspection=pd.DataFrame({'Actual':y_test, 'Predicted':y_pred})

#join X_test with the new dataframe
inspection=pd.concat([X_test,inspection], axis=1)

inspection.sample(20)
```

| | V1 | V2 | V7 | V8 | V9 | Actual | Predicted |
|---|---|---|---|---|---|---|---|
| 95 | 10 | 4 | 5 | 5 | 1 | 1 | 1 |
| 329 | 5 | 1 | 1 | 1 | 1 | 0 | 0 |
| 81 | 5 | 2 | 5 | 1 | 1 | 1 | 0 |
| 189 | 10 | 8 | 8 | 1 | 1 | 1 | 1 |
| 635 | 1 | 1 | 3 | 1 | 1 | 0 | 0 |
| 307 | 6 | 1 | 1 | 1 | 1 | 0 | 0 |
| 601 | 5 | 1 | 3 | 1 | 1 | 0 | 0 |
| 255 | 1 | 1 | 1 | 1 | 7 | 0 | 0 |
| 429 | 5 | 1 | 2 | 3 | 1 | 0 | 0 |
| 134 | 1 | 5 | 7 | 10 | 1 | 1 | 1 |
| 569 | 4 | 1 | 3 | 1 | 1 | 0 | 0 |
| 573 | 1 | 1 | 3 | 1 | 1 | 0 | 0 |
| 391 | 4 | 2 | 2 | 1 | 1 | 0 | 0 |

# View probabilities instead of predictions

```
#get predicted probabilities for the main class
y_pred_probs = logreg.predict_proba(X_test)
y_pred_probs = y_pred_probs[:, 1]
```

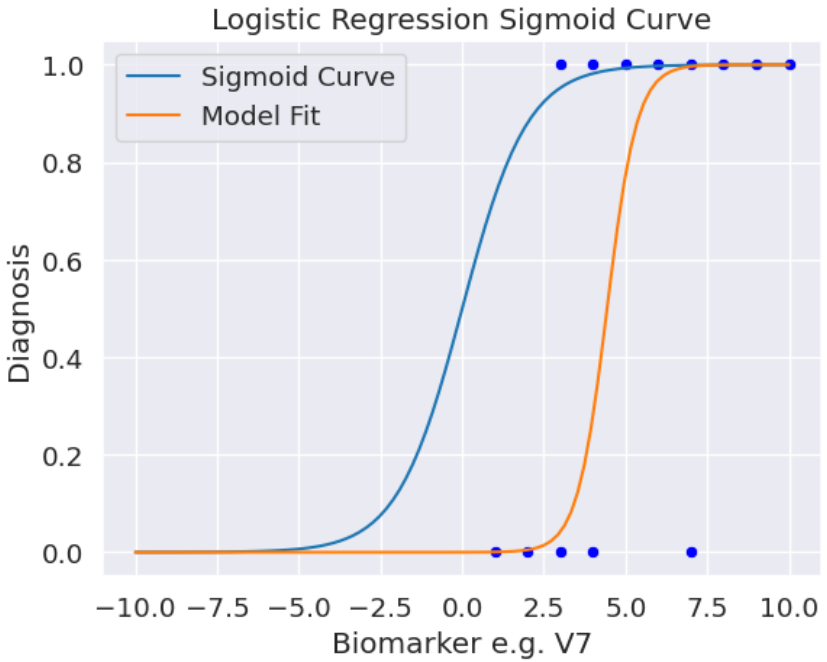
| | V1 | V2 | V7 | V8 | V9 | Actual | Predicted | Probability |
|---|---|---|---|---|---|---|---|---|
| 274 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0.000549 |
| 246 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0.000549 |
| 412 | 1 | 1 | 2 | 3 | 1 | 0 | 0 | 0.003237 |
| 367 | 4 | 1 | 1 | 1 | 1 | 0 | 0 | 0.002002 |
| 267 | 2 | 1 | 1 | 1 | 1 | 0 | 0 | 0.000845 |
| 8 | 8 | 4 | 2 | 5 | 2 | 0 | 0 | 0.154649 |
| 668 | 3 | 1 | 3 | 1 | 1 | 0 | 0 | 0.018595 |
| 77 | 8 | 7 | 5 | 5 | 4 | 1 | 1 | 0.947708 |
| 386 | 1 | 1 | 1 | 1 | 8 | 0 | 0 | 0.000592 |
| 698 | 4 | 4 | 7 | 3 | 1 | 0 | 1 | 0.943593 |
| 304 | 5 | 1 | 1 | 1 | 1 | 0 | 0 | 0.003081 |
| 187 | 5 | 4 | 8 | 10 | 1 | 1 | 1 | 0.997813 |
| 269 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0.000549 |
| 231 | 10 | 10 | 10 | 3 | 1 | 1 | 1 | 0.999974 |

```
#join unseen y_test with predicted value and probability
inspection=pd.DataFrame({'Actual':y_test, 'Predicted':y_pred, 'Probability':y_pred_probs})

#join X_test with the new dataframe
inspection=pd.concat([X_test,inspection], axis=1)
inspection.sample(20)
```

# Model Visualisation

```
# Calculate the corresponding y values using
the model coefficients
coef = logreg.coef_.flatten()
intercept = logreg.intercept_
```

```
print('Diagnosis= ', '%.3f' % intercept, '+', '%.3f'
%coef[0], '*V1', '+', '%.3f' %coef[1], '*V2', '+',
'%.3f' %coef[2], '*V7', '+', '%.3f' %coef[3], '*V8',
'+', '%.3f' %coef[4], '*V9')
```




**Diagnosis_logit= -9.694 + 0.432 *V1 + 0.186 *V2 + 1.339 *V7 + 0.219 *V8 + 0.011 *V9**

$$\hat{y} = b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_n x_n$$

**Diagnosis=Sigmoid(Diagnosis_logit)**

# Performance Metrics

# Model evaluation

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} = 1 - Error$$

$$Error = \frac{FP+FN}{TP+FP+TN+FN} = 1 - Accuracy$$

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

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

Recall – True Positive Rate

True labels

| | Negative | Positive |
|---|---|---|
| False labels | **TN** | FP |
| True labels | FN | **TP** |

Predictions

**Table 8.2** Evaluation Measures

| Term | Definition | Calculation |
|---|---|---|
| Sensitivity | Ability to select what needs to be selected | TP/(TP+FN) |
| Specificity | Ability to reject what needs to be rejected | TN/(TN+FP) |
| Precision | Proportion of cases found that were relevant | TP/(TP+FP) |
| Recall | Proportion of all relevant cases that were found | TP/(TP+FN) |
| Accuracy | Aggregate measure of classifier performance | (TP+TN)/(TP+TN+FP+FN) |

*Kotu and Deshpande, 2019, chapter 4*

# Model evaluation

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

True labels

| | Negative | Positive |
|---|---|---|
| False labels | **TN** | FP |
| True labels | FN | **TP** |

Predictions

$$Precision(PRE) = \frac{TP}{TP + FP}$$

Useful in spam detection

True labels

| | Negative | Positive |
|---|---|---|
| False labels | **TN** | FP |
| True labels | FN | **TP** |

Predictions

$$Specificity = \frac{TN}{TN+FP}$$

True labels

| | Negative | Positive |
|---|---|---|
| False labels | **TN** | FP |
| True labels | FN | **TP** |

Predictions

# Model evaluation

**Sensitivity – Recall (REC) – True Positive Rate (TPR)**

$$TPR = REC = \frac{TP}{TP+FN} = \frac{TP}{P}$$

True labels

| | Negative | Positive |
|---|---|---|
| False labels | **TN** | FP |
| True labels | FN | **TP** |

Predictions

**False Positive Rate (FPR)**

$$FPR = \frac{FP}{FP+TN} = \frac{FP}{N} = 1 - Specifity$$

True labels

| | Negative | Positive |
|---|---|---|
| False labels | **TN** | FP |
| True labels | FN | **TP** |

Predictions

Useful in imbalanced class problems such as cancer diagnosis – recall
Consider the domain problems

# Accuracy Paradox and F1 score

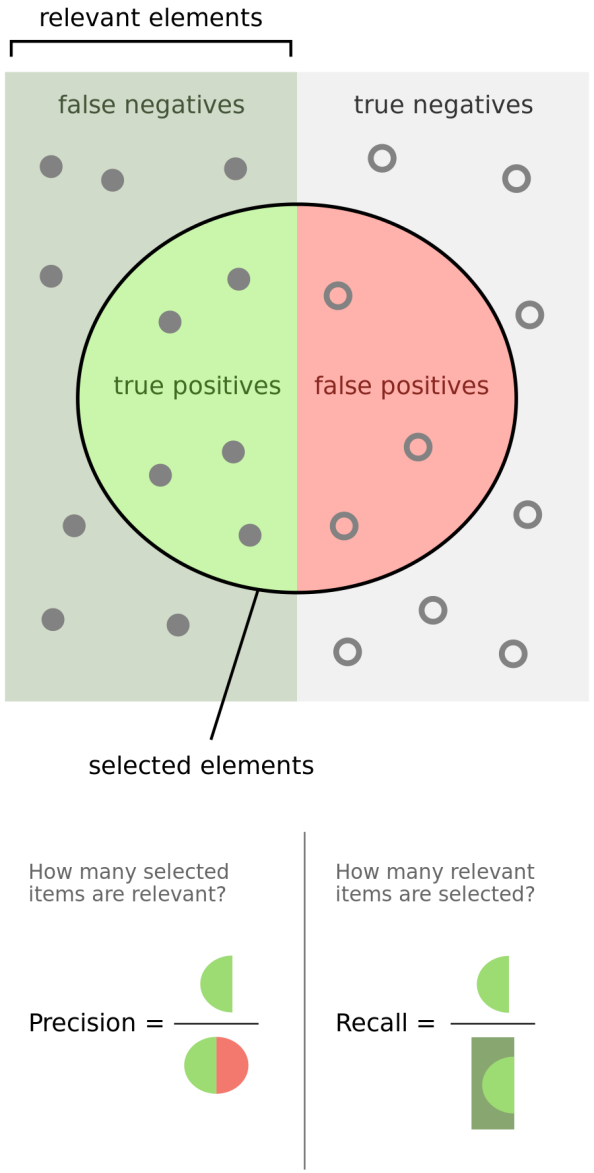
Problem: class imbalance but equal importance to precision and recall.

F1 score: harmonic mean of precision and recall:

$$F1 = 2 \times \frac{precision \times recall}{precision + recall}$$

Aim to get high F1, especially when FN and FP are important.

A good way to summarise the classification performance.



# Model performance evaluation in scikit-learn

```
#Model Evaluation, calculate metrics: Accuracy, Precision, Recall, F1,
print(f'Accuracy: {metrics.accuracy_score(y_test,y_pred): .3f}')
print(f'Precision: {metrics.precision_score(y_test,y_pred): .3f}')
print(f'Recall: {metrics.recall_score(y_test,y_pred): .3f}')
print(f'F1: {metrics.f1_score(y_test,y_pred): .3f}')
```

```
Accuracy: 0.9
Precision: 0.80
Recall: 0.88
F1: 0.84
```

| True labels | Negative | Positive |
|---|---|---|
| False labels | **TN** | FP |
| True labels | FN | **TP** |

Predictions

# Model evaluation

**Precision**

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

True labels

| | Negative | Positive |
|---|---|---|
| False labels | **TN** | FP |
| True labels | FN | **TP** |

Predictions

**Sensitivity – Recall (REC) – True Positive Rate (TPR)**

$$TPR = REC = \frac{TP}{TP+FN} = \frac{TP}{P}$$

True labels

| | Negative | Positive |
|---|---|---|
| False labels | **TN** | FP |
| True labels | FN | **TP** |

Predictions

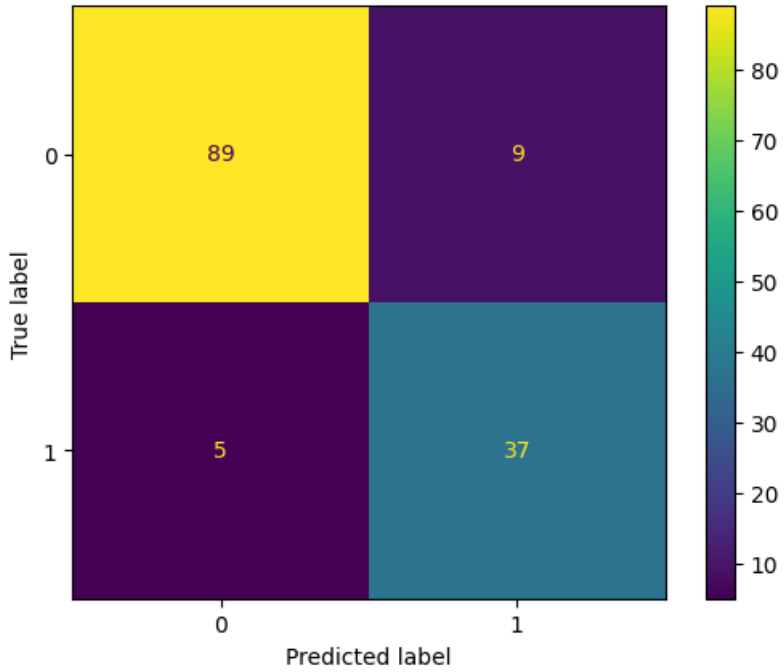
# Model performance evaluation in scikit-learn

| True labels | Negative | Positive |
|---|---|---|
| False labels | TN | FP |
| True labels | FN | TP |

Predictions

```
#print confusion matrix and evaluation report
from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
[[89  9]
 [ 5 37]]
              precision    recall  f1-score   support

           0       0.95      0.91      0.93        98
           1       0.80      0.88      0.84        42

    accuracy                           0.90       140
   macro avg       0.88      0.89      0.88       140
weighted avg       0.90      0.90      0.90       140
```



# What is the best threshold?

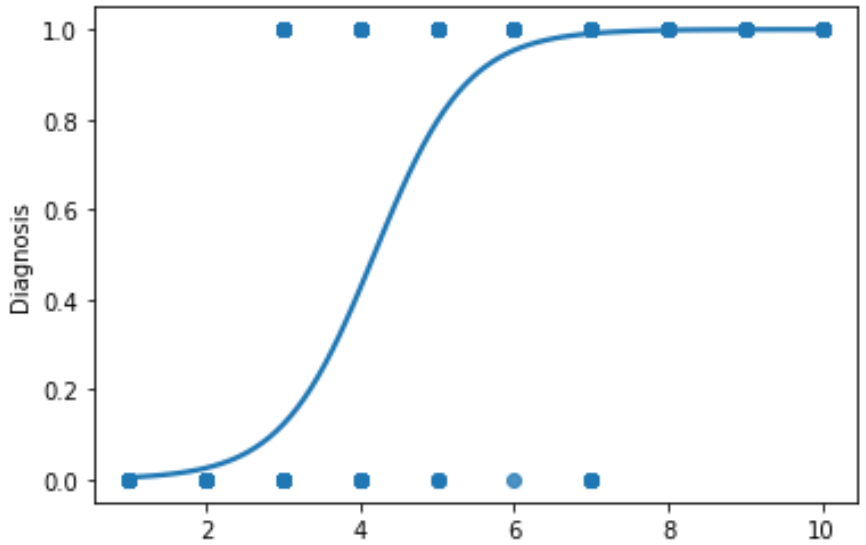
Recall that we convert

$$b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_n x_n$$

into $p \in \{0,1\}$

But we need to 'classify' p into classes, and assume:

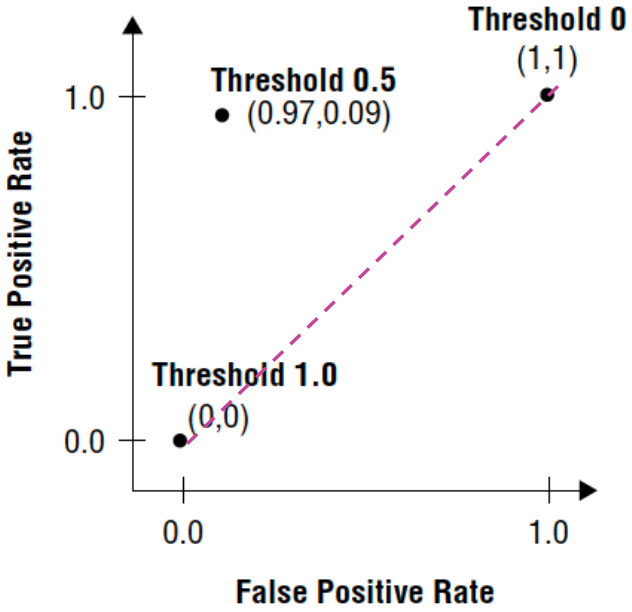
$\hat{y}$ { **1 if p> 0.5** (**is this the best threshold?**)
0 otherwise

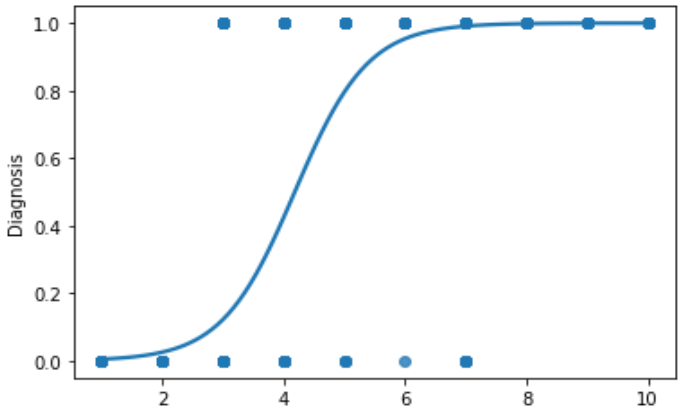


# Receiver Operating Characteristic (ROC) curve

## Plotting the TPR against the FPR at various threshold settings

0 = healthy
1 = cancerous




Adapted from Lee, 2019

Threshold = 1, all predictions: 0

| True labels | Negative | Positive |
|---|---|---|
| False labels | **TN** | FP=0 |
| True labels | FN | **TP=0** |

Predictions

Threshold = 0, all predictions: 1

| True labels | Negative | Positive |
|---|---|---|
| False labels | **TN = 0** | FP |
| True labels | FN = 0 | **TP** |

Predictions

# Model evaluation

**Sensitivity – Recall (REC) – True Positive Rate (TPR)**

$$TPR = REC = \frac{TP}{TP+FN} = \frac{TP}{P}$$

True labels

| | Negative | Positive |
|---|---|---|
| False labels | **TN** | FP |
| True labels | FN | **TP** |

Predictions

**False Positive Rate (TPR)**

$$FPR = \frac{FP}{FP+TN} = \frac{FP}{N}$$

True labels

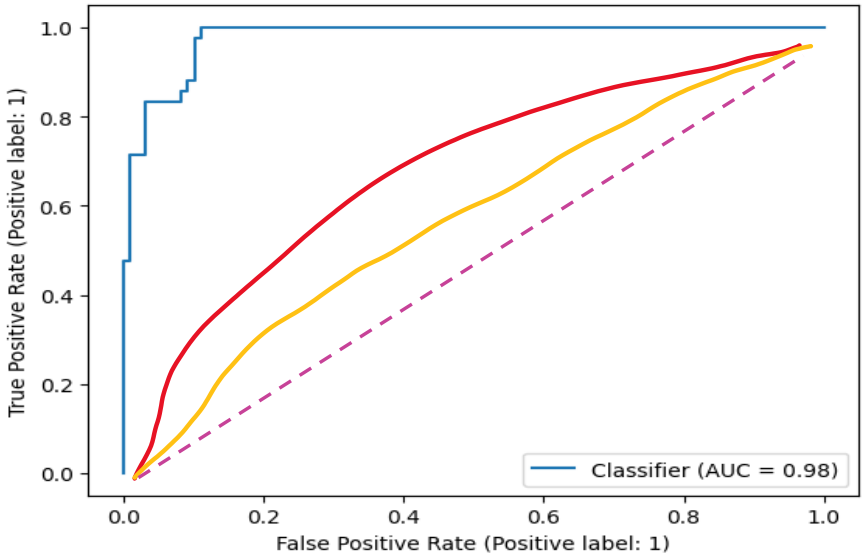
| | Negative | Positive |
|---|---|---|
| False labels | **TN** | FP |
| True labels | FN | **TP** |

Predictions

Useful in imbalanced class problems such as cancer diagnosis – recall
Consider the domain problems

# Receiver Operating Characteristic (ROC) curve

Plotting the TPR against the FPR at various threshold settings.

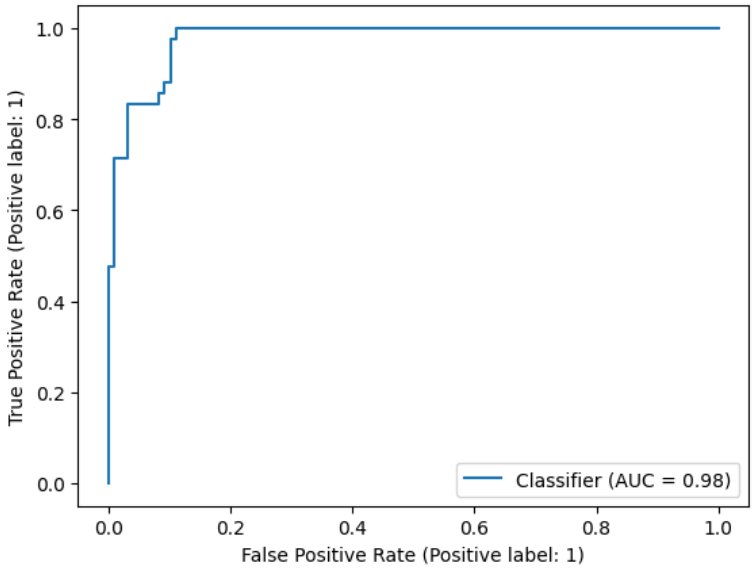


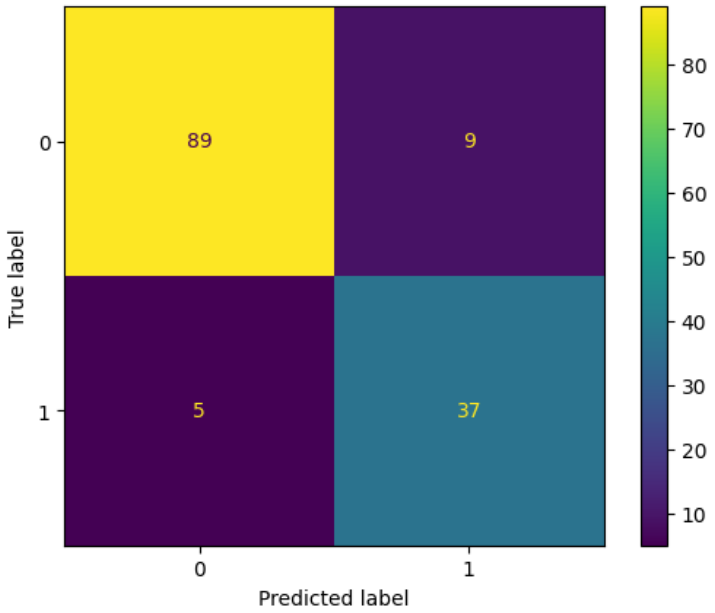

The area under the curve (AUC) is often considered as a summary of the model performance.

# Plot ROC curve and visualise Confusion matrix using built in functions

**NOTE: Multiple-class: One vs Rest (OvR) or One vs All (OvA)**

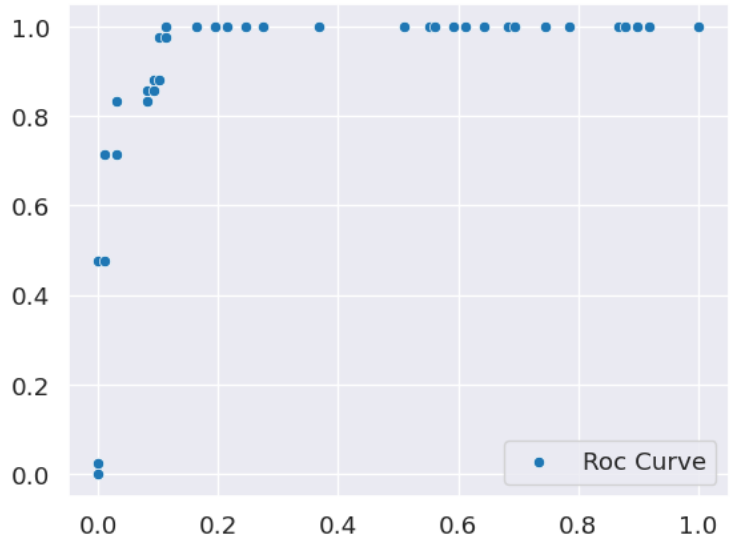
```
#import classes to display RocCurve and Confusion Matrix
from sklearn.metrics import RocCurveDisplay
from sklearn.metrics import ConfusionMatrixDisplay

RocCurveDisplay.from_predictions(y_test, y_pred_probs)
ConfusionMatrixDisplay.from_predictions(y_test, y_pred)
plt.show()
```

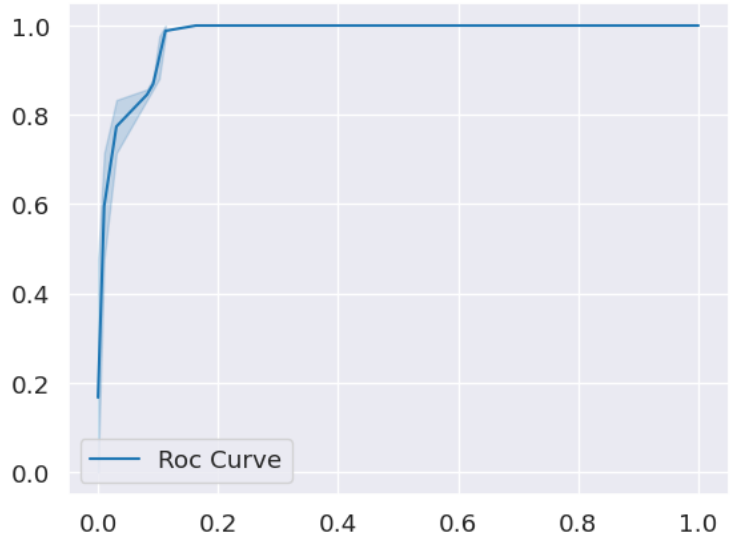


# Plot ROC curve

```
#get predicted probabilities for the primary label
y_pred_probs = logreg.predict_proba(X_test)
y_pred_probs = y_pred_probs [:, 1]


#get fpr and tpr and plot the ROC curve
from sklearn.metrics import roc_curve, roc_auc_score

lr_fpr, lr_tpr, thresholds = roc_curve(y_test, y_pred_probs)

sns.scatterplot(x=lr_fpr, y=lr_tpr, label='Roc Curve')


sns.lineplot(x=lr_fpr, y=lr_tpr, label='Roc Curve')
```



# What is the best the threshold?
## based on accuracy

```
# Calculate ROC curve and AUC
fpr, tpr, thresholds = roc_curve(y_test, y_pred_probs)

# initialize variables to store the best threshold and the highest accuracy
score
best_threshold = None
highest_ac_score = 0

# iterate over the thresholds and compute the accuracy score for each
for threshold in thresholds:
    y_pred_test = (y_pred_probs >= threshold).astype(int)
    ac_score = metrics.accuracy_score(y_test, y_pred_test)
if ac_score > highest_ac_score:
    highest_ac_score = ac_score
    best_threshold = threshold

# Assign predictions based on the new threshold
y_pred = (y_pred_probs >= best_threshold).astype(int)

print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

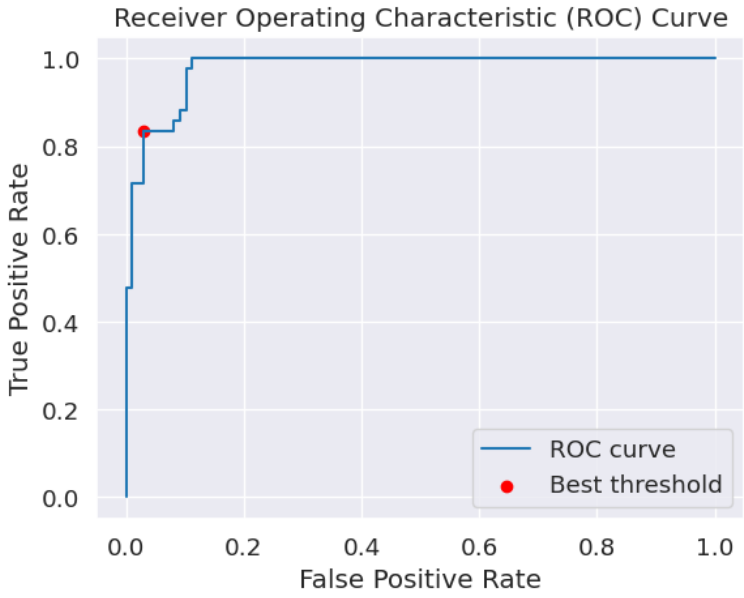
```
[[95  3]
 [ 7 35]]
              precision    recall  f1-score   support

           0       0.93      0.97      0.95        98
           1       0.92      0.83      0.88        42

    accuracy                           0.93       140
   macro avg       0.93      0.90      0.91       140
weighted avg       0.93      0.93      0.93       140
```

```
# print the best threshold and the highest AUC score on the test data
print('Best threshold:', best_threshold)
print('Highest accuracy score:', highest_ac_score)
```

```
Best threshold:
0.9143633687597816
Highest accuracy score:
0.9285714285714286
```

```
# Find the index corresponding to the specific threshold
best_index = (np.abs(thresholds - best_threshold)).argmin()
```

```
# plot the ROC curve and the best point
plt.plot(fpr, tpr, label='ROC curve')
plt.scatter(fpr[best_index], tpr[best_index], marker='o',
color='red', label='Best threshold')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend()
plt.show()
```



# Logistic regression

## Pros

- Explainable: Easy to interpret
- Visual representation
- Non-parametric: No assumptions on data distribution (linearity, normality)
- Less effort for data preparation, no need for normalisation
- Work for both numerical and categorical predictors

## Cons

- Target must be categorical, best with binary (dichotomous)
- Require large datasets. Overfitting if datasets are small
- Complex when having multi-class targets

# Assumptions

- The target should be categorical
- The datapoints are independent
- No extreme outliers
- No severe collinearity among the predictors
- There exists a linear relationship between each predictors and the logit of the target i.e. log(p / (1-p))
- Sample size is large enough

https://www.statology.org/assumptions-of-logistic-regression/

# Linear Regression and Logistic Regression

## Linear Regression

- **Supervised ML**
- **Linear Regression equations**

- Estimation: target is continuous
- Best-fit line

- Loss function: Prediction Error
- Cost function: Mean squared error

- Assume linear relationships between predictors and target

## Logistic Regression

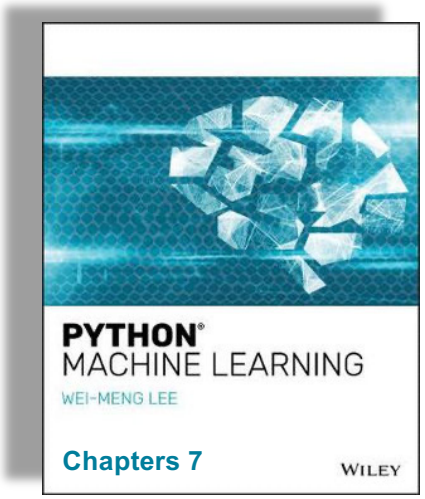
- **Supervised ML**
- **Linear Regression equations**

- Classification: target is categorical
- S-curve (Fit the regression values to the sigmoid curve)

- Lost function: maximum likelihood estimation
  Cost function: maximum likelihood estimation values

- Not assume linear relationships between predictors and target

# Useful readings




https://ebookcentral-proquest-com.ezproxyb.deakin.edu.au/lib/deakin/detail.action?docID=5747364

**AND** See useful sites in Lecture 4 and Lab 4.

Logistic Regression:
https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

Train Test Split:
https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html?highlight=train_test_split#sklearn.model_selection.train_test_split

Classification metrics:
https://scikit-learn.org/stable/modules/model_evaluation.html#classification-metrics