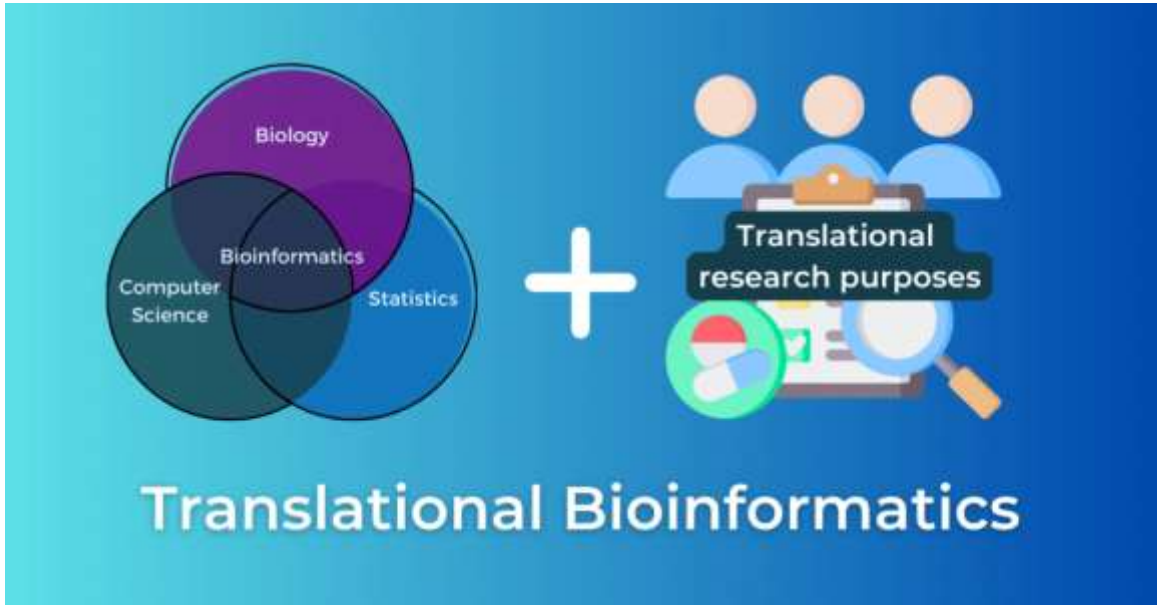
Data Science in Translational Bioinformatics: Foundations and Applications

Nidhi Malhotra
Assistant Professor
Department of Chemistry
School of Natural Sciences
Shiv Nadar Institution of Eminence, Delhi-NCR

Data Science in Translational Bioinformatics

- Bioinformatics is big data analysis for biological data.
- Translational Bioinformatics is defined as "the union of translational medicine and bioinformatics" that "makes it possible to access the knowledge of scientific evidence and apply it to clinic practice.
- Bioinformatics involves the **analysis of biological data for any purpose**, and the term 'bioinformatics' can also refer to the development of methods or software for understanding biological data. Translational bioinformatics,



https://www.fiosgenomics.com/translational-bioinformatics-advance-

Applications of Translational Bioinformatics

- **Drug discovery**: Identifies drug targets and designs new medications
- **Precision Medicine**: Uses individual genetic profiles to tailor treatments.
- **Pharmacogenomics**: studies how a person's genes affect their response to drugs
- **Other applications**: Drug Repurposing, Medical image analysis, clinical and disease research, Viroinformatics, Genetic Epidemiology

Data integration and hypothesis

Artificial Intell

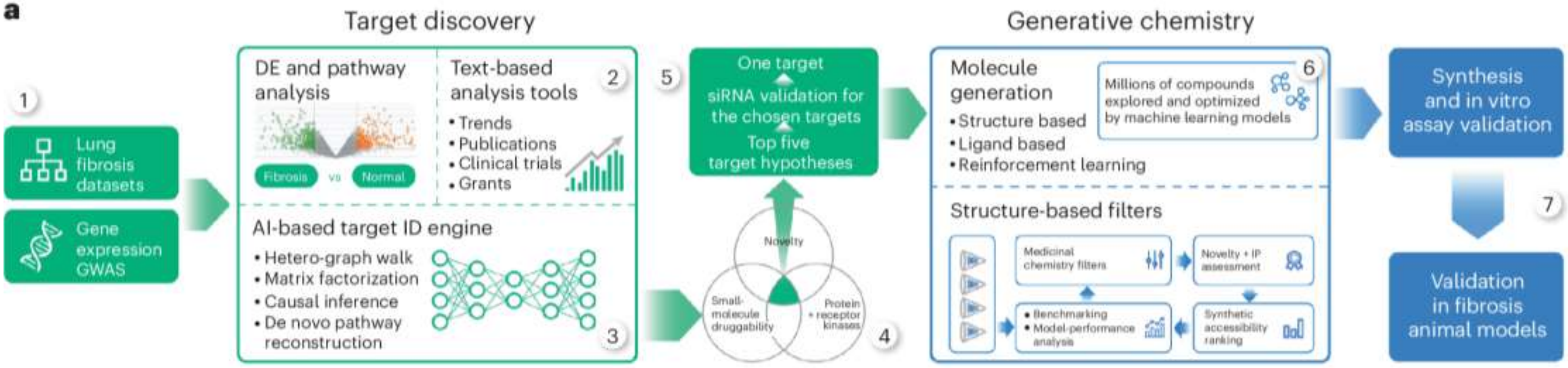
Learn patterns & build predictive

Such insights can improve diagnostics, inform clinical decision-making and accelerate drug development research.

Recent Breakthrough Example

Is AI hype in drug development about to turn into reality?

- Recent research led by **Alex Zhavoronkov**, founder and CEO of Insilico Medicine, a Hong Kong-based biotech company, has resulted in a breakthrough: the first drug fully generated by artificial intelligence has entered Phase IIa clinical trials.
- Rentosertib drug: designed to treat idiopathic pulmonary fibrosis (IPF), a fatal lung disease.
- AI methodology used both in target identification (TNIK, identified using PandaOmics) and drug development (Chemistry42 generative engine)

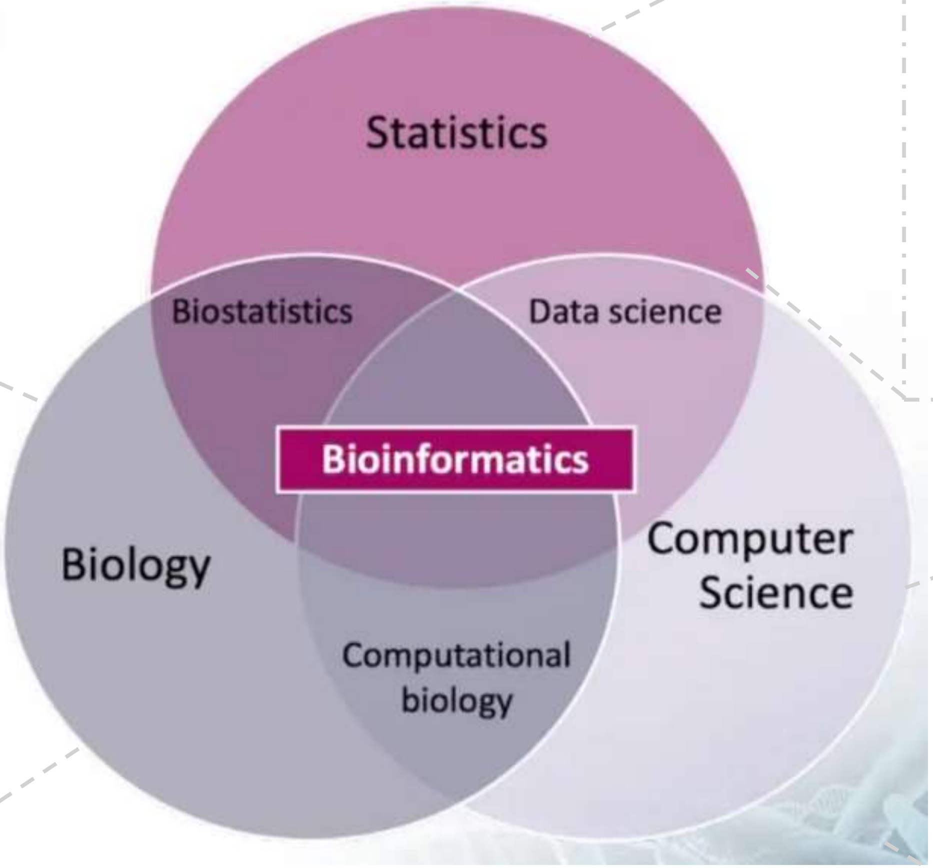


Ren, F., Aliper, A., Chen, J. *et al.* A small-molecule TNIK inhibitor targets fibrosis in preclinical and clinical models. *Nat Biotechnol* **43**, 63–75 (2025)
https://360info.org/is-ai-hype-in-drug-development-about-to-turn-into-reality/

Basics To Know Before Starting Translational Bioinformatics

Databases with clinical information

- ❑ Gene Expression
- ❑ Differential gene expression
- ❑ Biological Pathways
- ❑ Drug repurposing/designing/sens



Data Analysis & Interpretation:

- ❑ Regression
- ❑ Classification

Programming Languages:

- ❑ Python
- ❑ R
- ❑ C/C++

Algorithms

Introduction to Python

- Python syntaxes are simple, easy to read and learn. It has automatic memory management and is an open source.

- It is better suitable for machine learning, and large-scale web applications.

- It comes with build-in data structures, and has many libraries that are necessary to carry out major science-related functions.

- Useful libraries in python:

  - Pandas: For analyzing structured data, can aid in cleaning, transformation & analysis of big datasets
  - NumPy: Essential for performing numerical operations
  - Matplotlib: To visualize data/plotting graphs
  - Sklearn: For machine-learning and data-mining algorithms

- Integrated Development Environment (IDE) for python: PyCharm, Jupyter, V S code

- Google colab is a web-based Jupyter notebook environment provided by Google. It lets you write, run and share python code right in your browser, with no installation needed.

Basic Python Concepts

Classes and Objects

A·class is like a template or blueprint that defines the structure and behavior of something.
It specifies:

- Attributes (properties/data)
- Methods (actions/functions)

An object is an instance of a class — a real, usable thing created using the blueprint.

❑ Integer object:

```
x = 5
print(type(x))
```

Output: <class ‘int’>

❑ String object:

```
text = “hello”
print(type(text))
```

Output: <class ‘str’>

Basic Python Concepts

- **Lists object:** c = [1,2,3] # list object

- **Dataframe object:** 2D table-like data structure (rows & columns)

- **Dictionary:**

A **dictionary** in Python is a data type that stores information in **key–value pairs** — just like a real dictionary!

Word (Key)	Meaning (Value)
"Cat"	"A small animal"
"Book"	"A collection of pages"

Code

```
my_dict = {
    "Cat": "A small animal",
    "Book": "A collection of pages"
}
print(my_dict["Cat"])
```

Output

```
A small animal
```

Basic Python Concepts

Functions and library:

- Let's say we need to add two numbers:
  - a=3, b=2; a+b=5

Functions:

```
def add_numbers(a, b):
    return a + b

print(add_numbers(5, 7))   # Output: 12
```

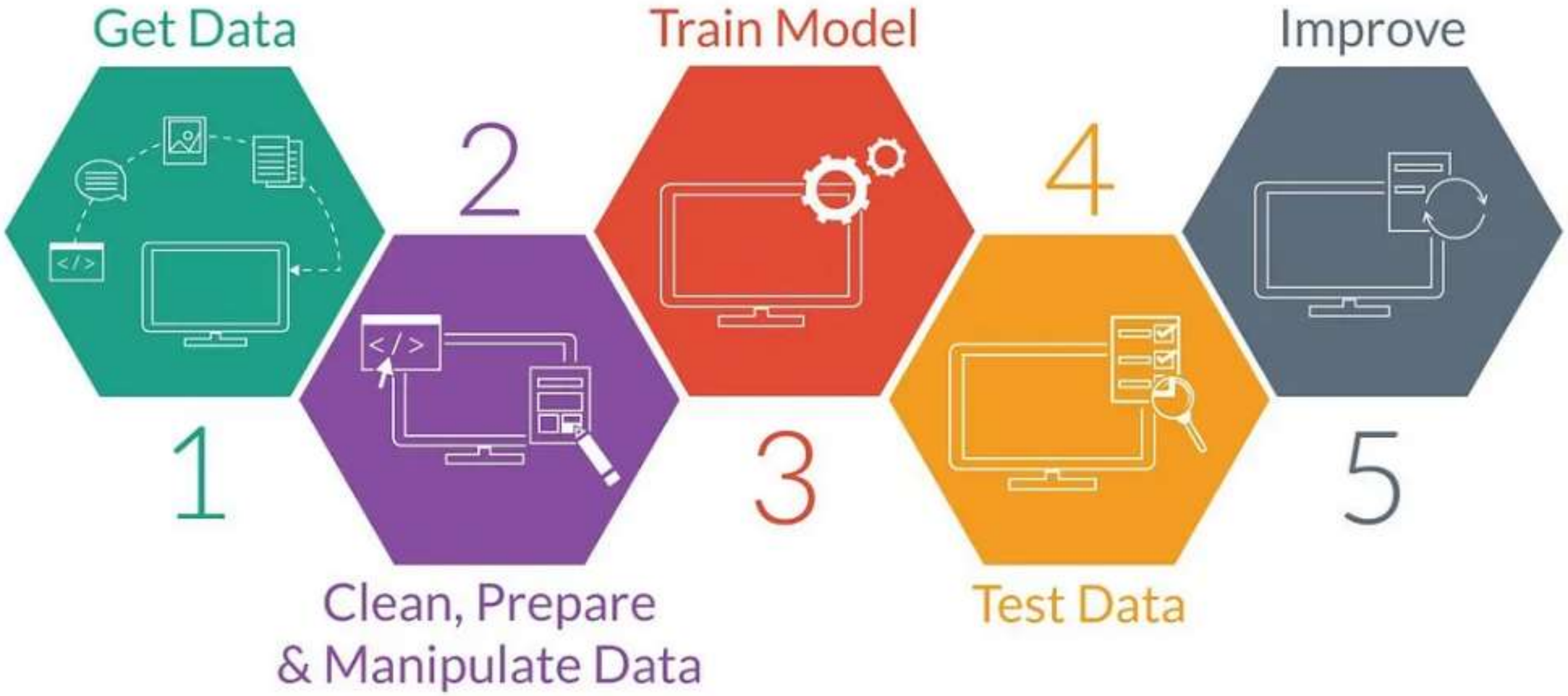
```
(): Do something (call a function)
[]: Get something (access element)
```

Library:

```
import math
print(math.fsum([5, 7]))   # Output: 12.0
```

```
library.function(attributes)
                     input
```

Machine Learning Steps



Biological Application:
Predicting cancer cell line drug sensitivity

Aim:

Predict whether a cancer cell line is sensitive or resistant to a given drug based on two gene expression features

CellLine	GeneA_Expression	GeneB_Expression	Drug_Response
CL1	8.1	4.3	Sensitive
CL2	6.5	3.0	Resistant
CL3	9.0	5.1	Sensitive
CL4	7.2	2.8	Resistant
CL5	NaN	4.5	Sensitive
CL6	5.8	NaN	Resistant

Understanding cancer cell line, gene expression and drug sensitivity

Cancer Cell Lines

Definition: Lab-grown cells derived from tumors, used as a model system to study cancer biology.
Purpose: Mimic human cancer for experiments without testing directly on patients.
Example: HeLa (cervical cancer), MCF-7 (breast cancer)

Gene Expression

Definition: The process by which information from a gene is used to make proteins.
Why it matters: Levels of gene expression can influence how cancer cells behave or respond to drugs.
Example: High expression of *EGFR* may make a tumor sensitive to EGFR inhibitors.

Drug Sensitivity

Definition: How responsive a cancer cell line is to a drug.
Sensitive: Drug effectively kills or inhibits the cells.
Resistant: Cells survive or continue growing despite drug treatment.
Measurement: Often quantified as IC50 (drug concentration that inhibits 50% of cells).

Lower IC50 → More sensitive cells
Higher IC50 → More resistant cells

Step 1: Get the data

Dictionary

Data Frame (structured, easy to read, has built-in tools for analysis)

```
import pandas as pd

# Create data
data = {
    'CellLine': ['CL1', 'CL2', 'CL3', 
'CL4', 'CL5', 'CL6'],
    'GeneA_Expression': [8.1, 6.5, 
9.0, 7.2, None, 5.8],
    'GeneB_Expression': [4.3, 3.0, 
5.1, 2.8, 4.5, None],
    'Drug_Response': ['Sensitive', 
'Resistant', 'Sensitive', 'Resistant', 
'Sensitive', 'Resistant']
}
df = pd.DataFrame(data)
```

Object.Method(argument)
Or
Library.Function(argument)

Step 2: Clean, Prepare and Manipulate data

Clean the data

Fill missing values with mean

```
df =
df.fillna(df.mean(numeric_only=
True))
```

```
print(df)
```

Manipulate the data

Convert categorical labels (“Sensitive”/”Resistant”) into numeric values for ML

```
df['Drug_Response'] =
df['Drug_Response'].map({'Sensitive': 1,
'Resistant': 0})
```

	CellLine	GeneA_Expression	GeneB_Expression	Drug_Response
0	CL1	8.10	4.30	1
1	CL2	6.50	3.00	0
2	CL3	9.00	5.10	1
3	CL4	7.20	2.80	0
4	CL5	7.32	4.50	1
5	CL6	5.80	3.94	0

Step 3a: Split into Train and Test Sets

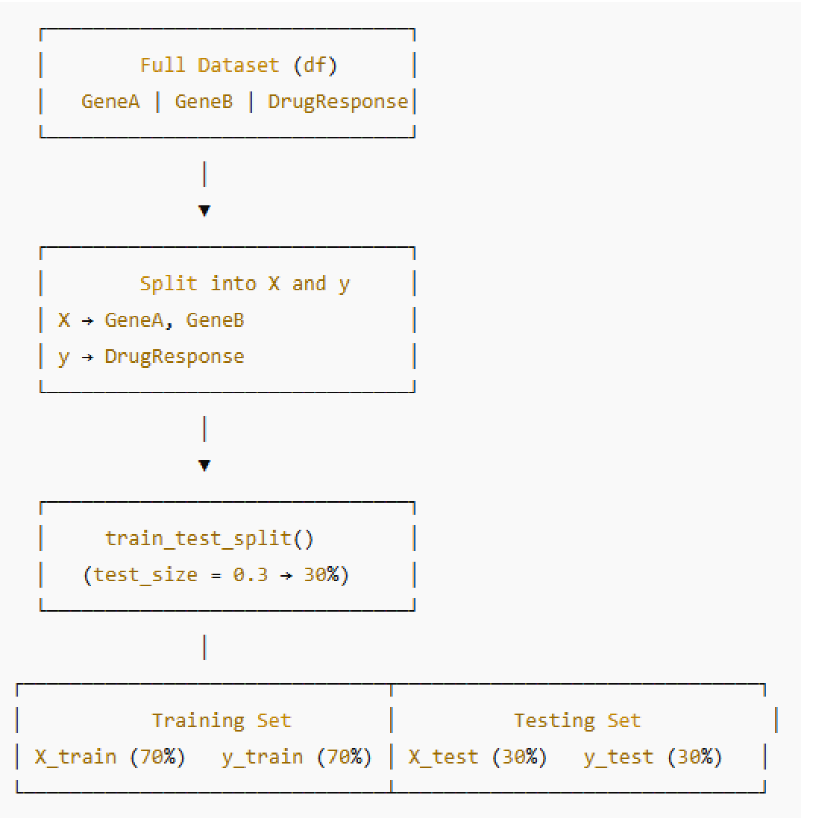
```
from sklearn.model_selection import
train_test_split


X = df[['GeneA_Expression',
'GeneB_Expression']]
y = df['Drug_Response’]

X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.3,
random_state=42)
```

X=> Features/In

y=> Target/Outp

70% data for traini
30% for testing

Note: cell lines are labels/identifier that doesn’t contain biological or numerical information that influences the prediction



Step 3a: Train the model

Logistic Regression

```
from sklearn.linear_model import
LogisticRegression

model = LogisticRegression()
model.fit(X_train, y_train)
```

Finds the best mathematical relationship between gene expressions (X) and drug response (y)

$$p = \frac{1}{1 + e^{-(b_0 + b_1 x_1 + b_2 x_2)}}$$

If $p \geq 0.5$ → classify as **Sensitive (1)**

If $p < 0.5$ → classify as **Resistant (0)**

Where:

- p = probability that the cell line is *Sensitive*
- x_1, x_2 = gene expression values (GeneA, GeneB)
- b_0 = intercept (constant term)
- b_1, b_2 = coefficients (weights learned for each gene)

Term	Meaning
x_1, x_2	Gene expression levels
b_0	Intercept (baseline sensitivity)
b_1, b_2	Effect of each gene
Equation	$p = 1/(1 + e^{-(b_0 + b_1 x_1 + b_2 x_2)})$
Output	Probability of being "Sensitive"

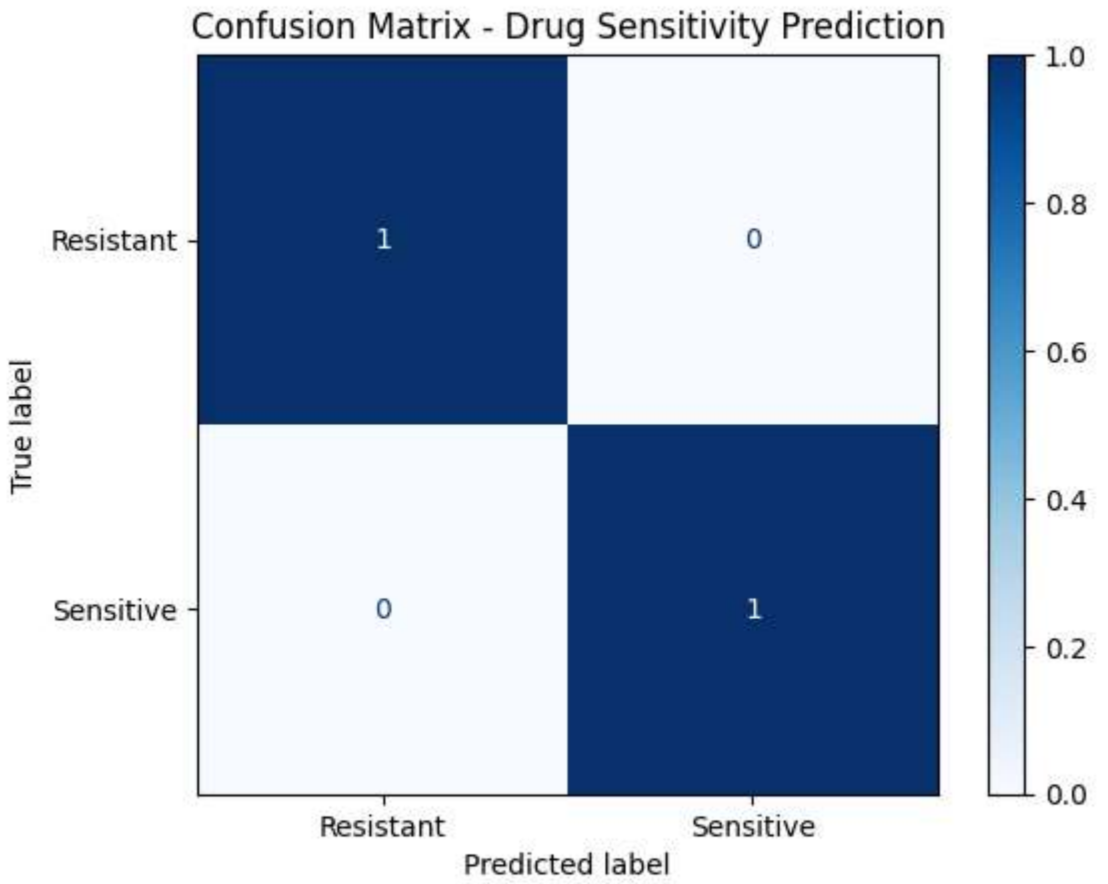
Step 4: Test and Evaluate the model

Test the model

```
y_pred = model.predict(X_test)
```

Evaluate performance

```
from sklearn.metrics import accuracy_scor
confusion_matrix, ConfusionMatrixDisplay
import matplotlib.pyplot as plt

print("Accuracy:", accuracy_score(y_test,
y_pred))

cm = confusion_matrix(y_test, y_pred)
disp =
ConfusionMatrixDisplay(confusion_matrix=c
display_labels=["Resistant", "Sensitive"]
disp.plot(cmap="Blues")
plt.title("Confusion Matrix - Drug
Sensitivity Prediction")
```



Accuracy = 1

Evaluating model: Confusion Matrix

	Predicted No	Predicted Yes
Actual No	45 True Negative (TN)	5 False Positive (FP)
Actual Yes	5 False Negative (FN)	95 True Positive (TP)

Accuracy: Overall, how often is the classifier correct?

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

Precision: When it predicts yes, how often is it correct?

$$Precision = \frac{TP}{Predicted\,Yes}$$

True Positive Rate: When it's actually yes, how often does it predict yes?

Sensitivity/Recall

$$True\ Positive\ rate = \frac{TP}{Actual\,Yes}$$

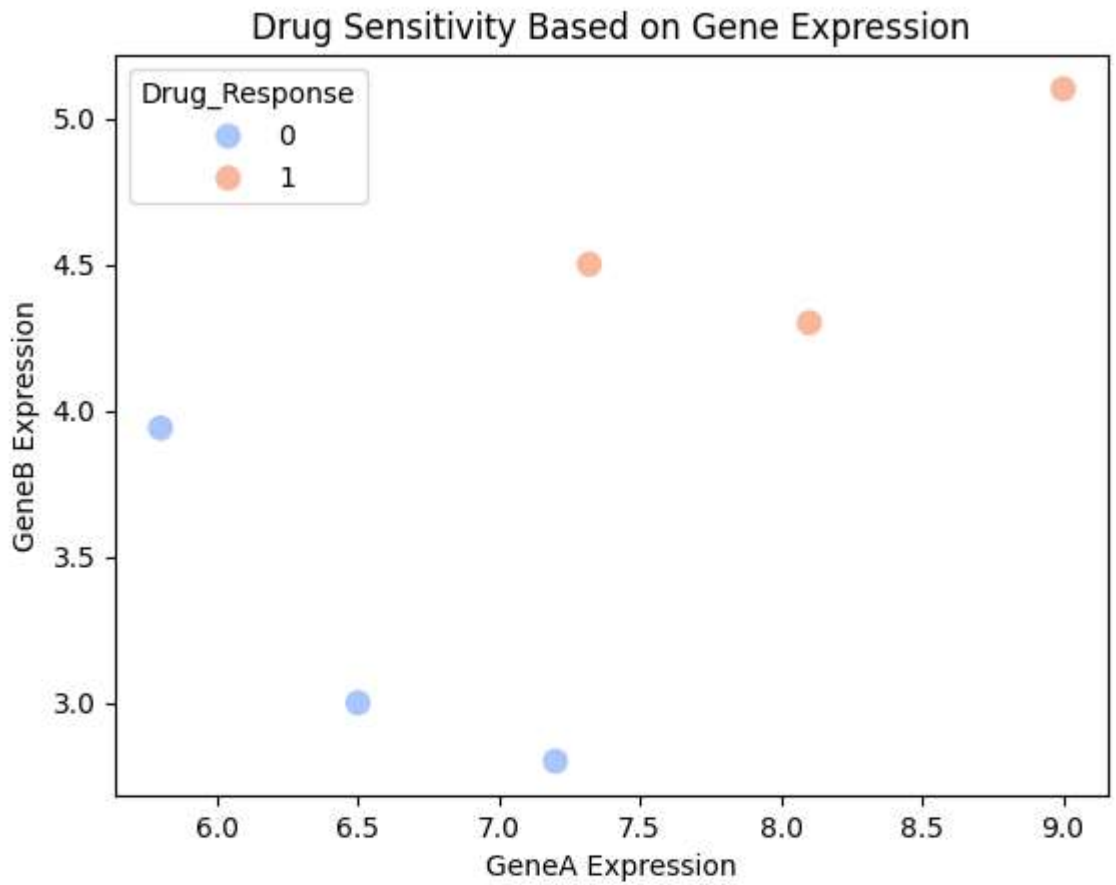
True Negative Rate: When it's actually no, how often does it predict no?

also known as "Specificity"

$$True\ Negative\ rate = \frac{TN}{Actual\,No}$$

Visualize Feature Relationships

```
import seaborn as sns

sns.scatterplot(data=df,
x='GeneA_Expression', y='GeneB_Expression',
                hue='Drug_Response',
palette='coolwarm', s=100)
plt.title("Drug Sensitivity Based on Gene
Expression")
plt.xlabel("GeneA Expression")
plt.ylabel("GeneB Expression")
plt.show()
```



Step 5: Model improvisation can include adding more

Real World Examples using big datasets

1) The Cancer Genome Atlas Program (TCGA): https://www.cancer.gov/ccg/research/genome-sequencing/tcga
   - UALCAN: https://ualcan.path.uab.edu/analysis.html
   - GEPIA: http://gepia.cancer-pku.cn/

2) Gene Expression Omnibus (GEO): https://www.ncbi.nlm.nih.gov/geo/
   - Geo2R (Series GSE309506): https://www.ncbi.nlm.nih.gov/geo/geo2r/

3) Kaggle Database: https://www.kaggle.com/code/siborakauri/drug-sensitivity

4) Genomics of Drug Sensitivity in Cancer (GDSC): https://www.cancerrxgene.org/

https://drive.google.com/drive/folders/1TGHr1Xf42tYHETFYEcbd2uKw2a21rrWI?usp=sharing