# Advanced Machine learning Mastering Course

Introduced by

George Samuel

Innovisionray.com

**2024**

# Agenda


1 Machine Learning Foundations

2 Training Models

3 Testing Models

4 Evaluation Metrics

5 Detecting Errors

6 Putting it all together

7 Bayesian Learning

8 Titanic Survival Exploration

10 Predicting Boston Housing Prices

# Training Models

# Training Models

How well is my model doing?

- model good or not
- we are going to learn metrics to tell us if it is good or not

How do we improve the model bases on these metrics?

- we will learn to <u>evaluate</u> and <u>validate</u> and <u>improve</u>

problem

**Machine Learning problems**

Tools

**Regression- Logistic regression, SVM,Naïve….etc.**

Measurement Tools

**Accuracy, Recall,precesion,F1 score and ….etc.**

## Statistics Refresher

- we'll assume familiarity with concepts in statistics such as *mean*, *median*, *variance*, etc.

- If it's been a while since you learned these and you feel that you need a refresher, please check our free courses

1- Descriptive statistics
2- Inferential statistics.

Full part in
Week-3-4-5

## Pandas

- Loading data into Pandas

- In this section we'll overview the basics on how to open a dataset in Pandas.
- If you want to go more in detail, please check the Pandas Documentation…..we will explain in details later.
- For this, we'll be using the dataset called **2_class_data.csv** which looks as follows:

To open this dataset, which comes as a csv file, we'll use the command read_csv in Pandas:

```
import pandas
data = pandas.read_csv("file_name.csv")
```

# Training Models

## Numpy

- Now that we've loaded the data in Pandas.
- we need to split the input and output into Numpy arrays, in order to apply the classifiers in scikit learn.
- This is done in the following way: Say we have a pandas data frame called df, like the following, with four columns labeled A, B, C, D:

**data frame**

| | A | B | C | D |
|---|---|---|---|---|
| **0** | 1 | 2 | 3 | 4 |
| **1** | 5 | 6 | 7 | 8 |
| **2** | 9 | 10 | 11 | 12 |

If we want to extract column A, we do the following:

```
>> df['A']
   0    1
   1    5
   2    9
   Name: A, dtype: int64
```

**Now, if we want to extract more columns, we just need to specify them, as follows:**

```
>> df[['B', 'D']]
```

And the result is the following Data Frame:

| | B | D |
|---|---|---|
| **0** | 2 | 4 |
| **1** | 6 | 8 |
| **2** | 10 | 12 |

And finally, we turn these pandas Data Frames into NumPy arrays. The command for turning a Data Frame df into a NumPy array is very simple:

```
>> numpy.array(df)
```

```
import pandas as pd
import numpy as np

data = pd.read_csv("data.csv")

# TODO: Separate the features and the labels into arrays called X and y

X = np.array(data[['x1', 'x2']])
y = np.array(data['y'])
```

# scikit learn

most important classification algorithms in Machine Learning, including the following:

- Logistic Regression
- Neural Networks
- Decision Trees
- Support Vector Machines

Now, we'll have the chance to use them in real data! In sklearn, this is very easy, all we do is define our classifier, and then use the following line to fit the classifier to the data (which we call X, y):

**Here are the main classifiers we define, together with the package we must import:**

```
classifier.fit(X,y)
```

# Training Models

## Logistic Regression

```
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression()
```

## Neural Networks

```
from sklearn.neural_network import MLPClassifier
classifier = MLPClassifier()
```

## Decision Trees

```
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier()
```

## Support Vector Machines

```
from sklearn.svm import SVC
classifier = SVC()
```

## Example: Logistic Regression

Let's do an end-to-end example on how to read data and train our classifier. Let's say we carry our X and y from the previous section. Then, the following commands will train the Logistic Regression classifier:

```
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression()
classifier.fit(X,y)
```

# Training Models

## Quiz: Train your own model

Your goal is to use one of the classifiers above, between

- ❑ Logistic Regression,
- ❑ Decision Trees,
- ❑ Support Vector Machines
- ❑ Neural Networks are still not available in this version of sklearn, but we will be upgrading soon!), to see which one will fit the data better.

**Let's try to fit this data with an SVM Classifier, as follows:**

```
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import GradientBoostingClassifie
from sklearn.svm import SVC

# Logistic Regression Classifier
classifier = LogisticRegression()
classifier.fit(X,y)

# Decision Tree Classifier
classifier = GradientBoostingClassifier()
classifier.fit(X,y)

# Support Vector Machine Classifier
classifier = SVC()
classifier.fit(X,y)
```

- it seems that maybe we're not exploring all the power of an SVM Classifier. For starters, are we using the right kernel? We can use, for example, a polynomial kernel of degree 2, as follows:

```
>>> classifier = SVC(kernel = 'poly', degree = 2)
```

## Tuning Parameters Manually

So it looks like 2 out of 3 algorithms worked well last time, right? This are the graphs you probably got:

Logistic Regression

Decision Tree

Support Vector Machine

It seems that Logistic Regression didn't do so well, as it's a linear algorithm. Decision Trees managed to bound the data well (question: Why does the area bounded by a decision tree look like that?), and the SVM also did pretty well

**Now, let's try a slightly harder dataset, as follows:**

```
classifier = SVC(kernel = 'rbf', degree = 2, gamma = 200, C = None)
```

Let's try it ourselves, let's play with some of these parameters. We'll learn more about these later, but here are some values you can play with. (For now, we can use them as a black box, but they'll be discussed in detail during the Supervised Learning Section of this Course.)

- kernel (string): 'linear', 'poly', 'rbf'.
- degree (integer): This is the degree of the polynomial kernel, if that's the kernel you picked (goes with poly kernel).
- gamma (float): The gamma parameter (goes with rbf kernel).
- C (float): The C parameter.

In the quiz below, you can play with these parameters. Try to tune them in such a way that they bound the desired area! In order to see the boundaries that your model created, click on Test Run.

Note: The quiz is not graded. But if you want to see a solution that works, look at the solutions.py tab. The point of this quiz is not to learn about the parameters, but to see that in general, it's not easy to tune them manually. Soon we'll learn some methods to tune them automatically in order to train better models.

# Testing your models

Testing your models

## Which Model is better

**Regression Model :** is one that predicts a value like 4 -3 or 6.7 so we need line that fits the data closely if we have anew value on the X axis we approximate by finding the corresponding Y value over the line

**Classification Model :** when one wants to determine a state such as positive, negative or such as yes, no  or cat, dog

**Regression return a numeric value**

**Classification return a state**

# Testing your models

**How well is my model doing**

**Which Model is better**

Des not fit the data

Fit the data

How do we find a model that generalizes well?

**The answer of this question …this is where we introduce the concept of testing.**

# Testing your models

## Regression Model

TESTING

# Testing your models

## Classification Model

## Testing in SKlearn

## Evaluation Matrix

How well is my model doing?

### Example-1
### Medical Model

### Example-2

# Evaluation Metrics

## MEDICAL MODEL

When a patient is healthy and the model incorrectly diagnoses them as sick, this is also a mistake and it means we will be sending a healthy person for further examination or treatment

| | Diagnosed Sick | Diagnosed Healthy |
|---|---|---|
| Sick | True Positive | False Negative |
| Healthy | False Positive | True Negative |

# Evaluation Metrics

- Model that will help us detect a particular illness and tell if a patient is healthy or sick

## CONFUSION MATRIX

10, 000 PATIENTS

DIAGNOSIS

PATIENTS

| | Diagnosed Sick | Diagnosed Healthy |
|---|---|---|
| Sick | 1000 True Positives | 200 False Negatives |
| Healthy | 800 False Positives | 8000 True Negatives |

# Evaluation Metrics

- Model that will help us detect a spam detector which will help us determine if an email is spam or not

SPAM CLASSIFIER MODEL

CONFUSION MATRIX

1000 EMAILS

SPAM

EMAIL

| | Spam Folder | Inbox |
|---|---|---|
| Spam | 100 True Positives | 170 False Negatives |
| Not Spam | 30 False Positives | 700 True Negatives |

## Confusion Matrix Quiz

CONFUSION MATRIX

| | Guessed Positive | Guessed Negative |
|---|---|---|
| Positive | 6<br>True Positives | 1<br>False Negatives |
| Negative | 2<br>False Positives | 5<br>True Negatives |

How many True Positives, True Negatives, False Positives, and False Negatives, are in the model above?

# Evaluation Metrics

Type 1 and Type 2 Errors
Sometimes in the literature, you'll see False Positives and False Negatives as Type 1 and Type 2 errors. Here is the correspondence:

Type 1 Error (Error of the first kind, or False Positive): In the medical example, this is when we misdiagnose a healthy patient as sick.
Type 2 Error (Error of the second kind, or False Negative): In the medical example, this is when we misdiagnose a sick patient as healthy.

| | Guessed Positive | Guessed Negative |
|---|---|---|
| Positive | 6<br>True Positives | 1<br>False Negatives |
| Negative | 2<br>False Positives | 5<br>True Negatives |

## Accuracy

**How many did we classify correctly?**

One of the ways to measure how good a model is .

**The answer is, the ratio between the number of correctly classified points and the number of total points.**

| | Spam folder | Inbox |
|---|---|---|
| Spam | 100 | 170 |
| Not spam | 30 | 700 |

### Medical model

$$\text{Accuracy} = \frac{1000 + 8000}{10000} = 90\%$$

### Spam detector model

$$\text{Accuracy} = \frac{100 + 700}{1000} = 80\%$$

```
From sklearn.metrics import accuracy_score
```

# Quiz

- ACCURACY

Out of all the data, how many points did we classify correctly?

$$\text{Accuracy} = \frac{\text{Correctly classified points}}{\text{All points}}$$

$$= \frac{11}{11+3} = \frac{11}{14} = \mathbf{78.57\%}$$

# Evaluation Metrics

It's pretty tricky to just look at accuracy and use that to evaluate our model because it may completely miss the point when the data is skewed like this one.

## When accuracy won't work?

### Credit card fraud

Good

284335 Good transactions

Fraudulent

472 Fraudulent transactions

Accuracy ???

$$\text{Accuracy} = \frac{284335}{284887} = 99.83\%$$

This model must be pretty good if it's accurate is that high right?????

Try another model

Can we get a model that catches all the bad transaction?

I'm going to label all transactions fraudulent. So, that's great….right?

Not really

This model is not catching any of the bad ones and the point of the model Is to catch the fraudulent transactions

It's accidentally catching all the good ones.

Terrible model

# Evaluation Metrics

**What be worst?** **False Negatives and Positives**

## Quiz 1: The Medical Model

Question

In the medical example, what is worse, a False Positive, or a False Negative?

**False Positives**

✓ **False Negatives**

## Quiz 2: The Spam Model

Question

In the spam detector example, what is worse, a False Positive, or a False Negative?

✓ **False Positives**

**False Negatives**

## Precision and Recall

**Solution: False Positive and Negative**

So this model basically says "i don't really care if i find all the spam emails but one thing is for sure
if i say that an email is spam it better be a spam"

- A false negative means a spam email made its way into your inbox and all you had to do is delete it.

**So it's a bit of an inconvenience**

- On the other hand what's a false positive?

Your grandma the poor lady learn to type an email only to tell you she bake cookies and you delete it .

**that's terrible**

So here we want to punish false positives more than false negatives

**False negative is much worse than a false positive, since predicting that a sick person is healthy is much more dangerous than predicting that a healthy person is sick.**

For this we introduce two metrics,

**precision and recall**

they will measure the exact things that we want ,

- **the medical model need to be a high recall model**
- **the email model needs to be high precision model**

we can see the medical model and the email model are fundamentally different
the medical model as we saw is okay with false positive but not with false negative since we're okay misdiagnosing some patient as sick as long as we find all the sick people

the email model is okay with false negative but not with false positives since we don't necessarily need to find all the spam emails but one thing is for sure that if we do label something as spam it better be spam.

## Precision

so precision is this column because this column are the sick patient that we diagnose as sick

DIAGNOSIS

PATIENTS

| | Diagnosed Sick | Diagnosed Healthy |
|---|---|---|
| Sick | 1000 | 200 |
| Healthy | 800 | 9000 |

$$\text{precision} = \frac{1000}{1000+800}$$

$$= 55.6\%$$

$$\text{precision} = \frac{TP}{TP+FP}$$

Out of all the points predicted to be positive How many of them were actually positive?

Out of all the patients that we diagnosed as sick. How many were actually sick?

## Precision

Spam email model

FOLDER

EMAIL

| | Sent to Spam Folder | Sent to Inbox |
|---|---|---|
| Spam | 100 | 170 |
| Not Spam | 30 | 700 |

$$\text{precision} = \frac{100}{100+30}$$

$$= 76.9\%$$

Out od all the Emails….sent to the spam folder ----------how many were actually spam?

# Precision Quiz

- PRECISION

OUT OF THE POINTS WE HAVE PREDICTED TO BE POSITIVE, HOW MANY ARE CORRECT?

FOLDER

| EMAIL | Sent to Spam Folder | Sent to Inbox |
|---|---|---|
| Spam | True Positive | False Negative |
| Not Spam | False Positive | True Negative |

$$\text{precision} = \frac{6}{6+2} = 0.75$$

# Recall

**Out of the points that are labeled positive, How many of them were correctly predicted as positive?**

## For medical model

How many did we correctly diagnose as sick?

Very important measure on the medical field and they called as sensitivity.

DIAGNOSIS

PATIENTS

| | Diagnosed Sick | Diagnosed Healthy |
|---|---|---|
| Sick | 1000 | 200 |
| Healthy | 800 | 8000 |

How many of the positive points did I manage to catch?

**Recall**
**Is opposite**
Out of the patients that are sick, How many did we correctly diagnose as sick?

$$\text{Recall} = \frac{1000}{1000+200} = 83.3\%$$

# Recall

## For Email model

How many of them do we correctly send to spam folder?

**Remember**

We are worried about avoiding this X over here since we don't mind if we don't catch all the spam emails as long as the ones we caught are spam

FOLDER

| EMAIL | Sent to Spam Folder | Sent to Inbox |
|---|---|---|
| Spam | True Positive | False Negative |
| Not Spam | False Positive | True Negative |

$$\text{Recall} = \frac{100}{100+170} = 37\%$$

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

# Recall Quiz

- PRECISION

OUT OF THE POINTS WE HAVE PREDICTED TO BE POSITIVE, HOW MANY ARE CORRECT?

FOLDER

| EMAIL | Sent to Spam Folder | Sent to Inbox |
|---|---|---|
| Spam | 6 | 1 |
| Not Spam | 2 | 5 |

$$\text{Recall} = \frac{6}{6+1} = 0.857$$

# F1 Score

- $F_1$ SCORE

ONE SCORE?

MEDICAL MODEL

PRECISION: 55.7%

**RECALL: 83.3%**

AVERAGE=69.5%

SPAM DETECTOR

**PRECISION: 76.9%**

RECALL: 37%

AVERAGE=56.95%

Good Metric but I'm sure you're probable thinking in your head!!!!!!!!!!

that's not very different than accuracy that's not saying too much.

And the way to see how this average is not the best idea is to try it in the extreme example

**The question is**

**Do we want to be carrying two numbers around all the time?**

Do we want to be carry precision in one pocket and recall in the other one?

We kind of want to only have one score.

And I will be looking at the both any time we make a decision.

Again the question is ……How do we combine these two scores into one?

Can we think of a way?
Well, a pretty simple ways taking the average of precision and recall

# F1 Score

The question is
What is the precision and Recall of this model?

Precision=100%

Average=50%

Recall=0%

AVERAGE = 50.08%

PRECISION = 472/284,807 = 0.16%

# F1 Score

- HARMONIC MEAN

Y

ARITHMETIC MEAN = $\frac{x+y}{2}$

HARMONIC MEAN = $\frac{2xy}{x+y}$

X

PRECISION = 1
RECALL = 0
AVERAGE = 0.5

HARMONIC MEAN = 0

PRECISION = 0.2
RECALL = 0.8
AVERAGE = 0.5

HARMONIC MEAN = 0.32

~~ARITHMETIC MEAN (PRECISION, RECALL)~~

$F_1$ SCORE = HARMONIC MEAN (PRECISION, RECALL)

Precision Quiz

If the Precision of the medical model is **55.6%**, and the Recall is **83.3%**, what is the F1 Score?