Machine Learning and Optimization
Lecture 4

Supervised learning: classification – logistic regression

Professor Georgina Hall

Agenda for today

Learning objectives:

- Pitfalls of supervised machine learning
- Understand the difference between regression and classification
- Understand what logistic regression is
- Know some metrics relating to classification

How will we get there?

- Running example: default dataset
- Hands-on coding

INSEAD

Some pitfalls in supervised machine learning

Pitfalls in supervised ML (1/3)

Saw one already:

Overfitting

When the model is fitted too closely to the data

How do we get round it?

- This is the purpose of the validation/test sets
- If your model is performing well on the training set but not on the other sets, then you are overfitting
- If this is happening, start over

Pitfalls in supervised ML (2/3)

Train-test contamination

When the testing set serves to fine tune the model when it should only be used to evaluate it.

Examples of occurrence:

- If **data pre-processing** (imputation with the mean, scaling/ normalizing) is done on training and testing sets at the same time
- If **hyperparameters are tweaked** based on the testing set performance

Model contains testing set information: the performance observed on the testing set **does not reflect reality.**

Pitfalls in supervised ML (3/3)

Target leakage

When the features include information that wouldn't be available at time of prediction (typically based on the actual prediction)

got_pneumonia	age	weight	male	took_antibiotic_medicine	...
False	65	100	False	False	...
False	72	130	True	False	...
True	58	100	False	True	...

Source: Kaggle

How to avoid this?

Know your data and what your features represent!

INSEAD

Classification vs regression

An activity to start

Consider these questions:

- A transaction has just been made on a customer's card. Using the user's past transactions and the location of the transaction, can I determine whether the transaction was fraudulent or not?
- A patient has just turned up at the hospital you work in. (S)he has a set of symptoms. Can I determine whether the patient is suffering from meningitis or covid or the flu?

Think about the following question:

Would regression work to answer one, both or none of these questions?

Why/why not?

Classification vs regression (1/2)

INSEAD

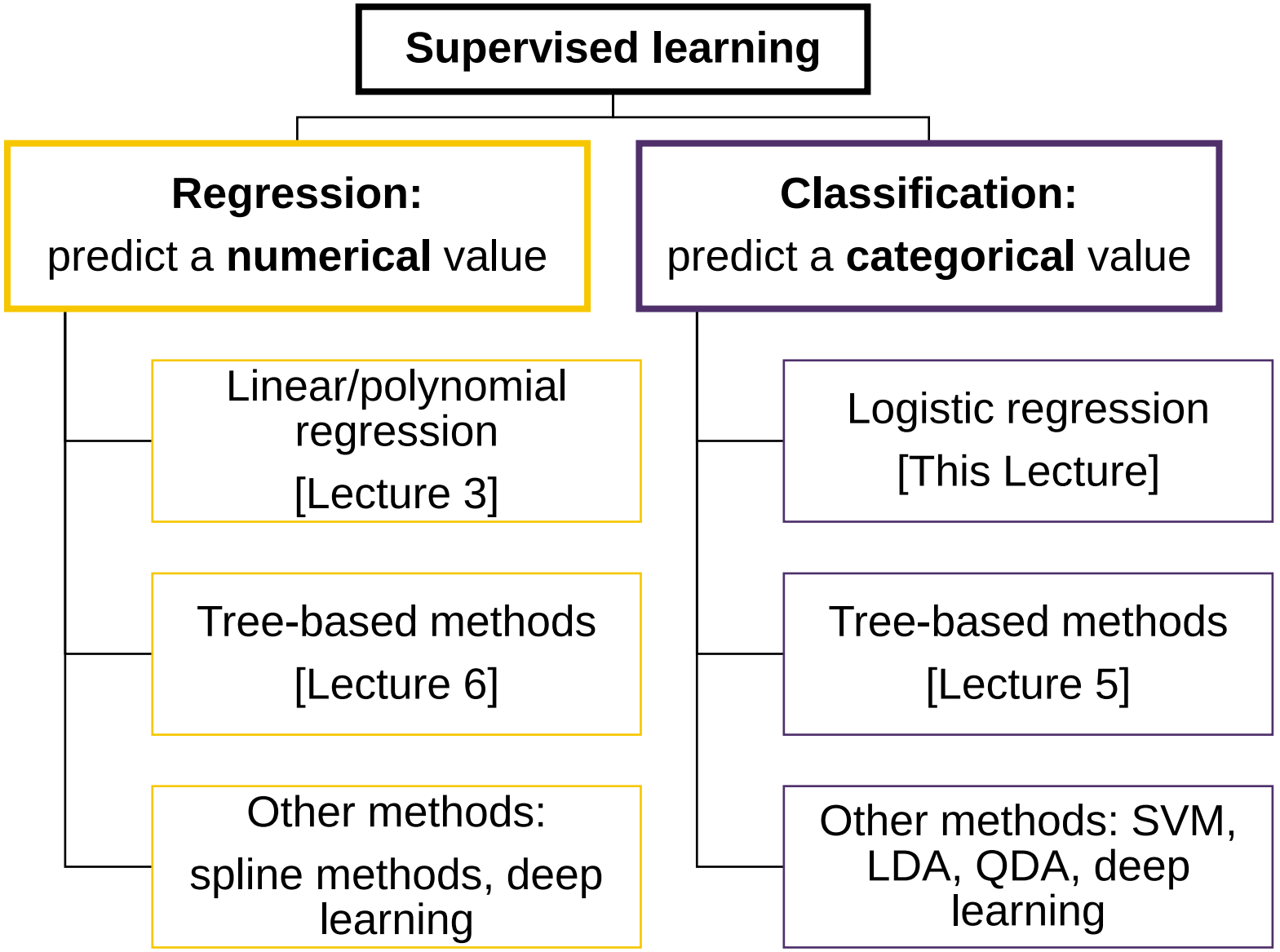
Regression

- **Input :** (x_i, y_i) with x_i =features and y_i =label
- **Goal:** given x_i, predict y_i
- Key fact: y_i here is a **number**

Classification

- **Input :** (x_i, y_i) with x_i =features and y_i =label
- **Goal:** given x_i, predict y_i
- Key fact: y_i here is a **categorical variable**
  yes or no: binary classification
  meningitis or flu or covid: multinomial classification

Classsification vs regression (2/2)



INSEAD

From linear to logistic regression

Banking (and defaulting) in the US

Beginning of Month 1: a credit card is acquired

Month 1: purchases are made with the credit card

End of Month 1: a balance has accumulated on the credit card:
⇒ Make the **minimum payment**: no impact on the credit score, interest on the amount outstanding
⇒ Make the **full payment**: no interest
⇒ Pay **nothing:** you become delinquent, impacts your credit score

End of Month 6: if no payments have been made, your account becomes a default account.

Taking a look at the Default dataset (1/3)

Download:

- ML&O Lecture 4 Exercise Book.ipynb
- Default.csv

Complete Part 1!

1) What are the features here? What do you think the label is? How many observations?

```
Default.head()
```

	default	student	balance	income
0	No	No	729.526495	44361.62507
1	No	Yes	817.180407	12106.13470
2	No	No	1073.549164	31767.13895
3	No	No	529.250605	35704.49394
4	No	No	785.655883	38463.49588

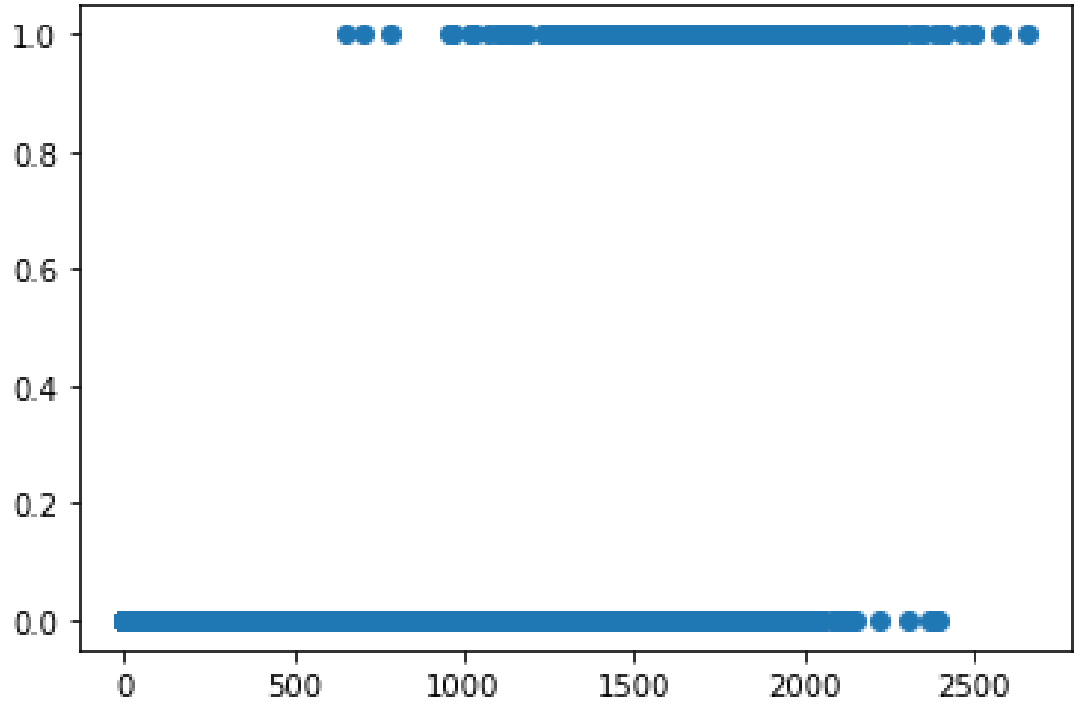
Taking a look at the Default dataset (2/3)

2) -3) Are there any duplicates? Missing values? Do the values taken by the dataset make sense? Return a fully numeric dataset.

	balance	income	student_Yes	default_Yes
0	729.526495	44361.62507	0	0
1	817.180407	12106.13470	1	0
2	1073.549164	31767.13895	0	0
3	529.250605	35704.49394	0	0
4	785.655883	38463.49588	0	0
...	...	...	...	...
9995	711.555020	52992.37891	0	0
9996	757.962918	19660.72177	0	0
9997	845.411989	58636.15698	0	0
9998	1569.009053	36669.11236	0	0
9999	200.922183	16862.95232	1	0

Taking a look at the Default dataset (3/3)

4) Use plt.scatter to plot default_Yes as a function of balance.



We’re going to move forward with a linear regression with just balance to get some intuition…

From linear to logistic regression (1/3)

- What are we taking as X and Y here?
- **Start on Part 2, Q1**. What is R^2?

```
X=Default[["balance"]]
Y=Default["default_Yes"]

lm = LinearRegression().fit(X, Y)

print(lm.intercept_)

print(lm.coef_)

print(lm.score(X,Y))
```

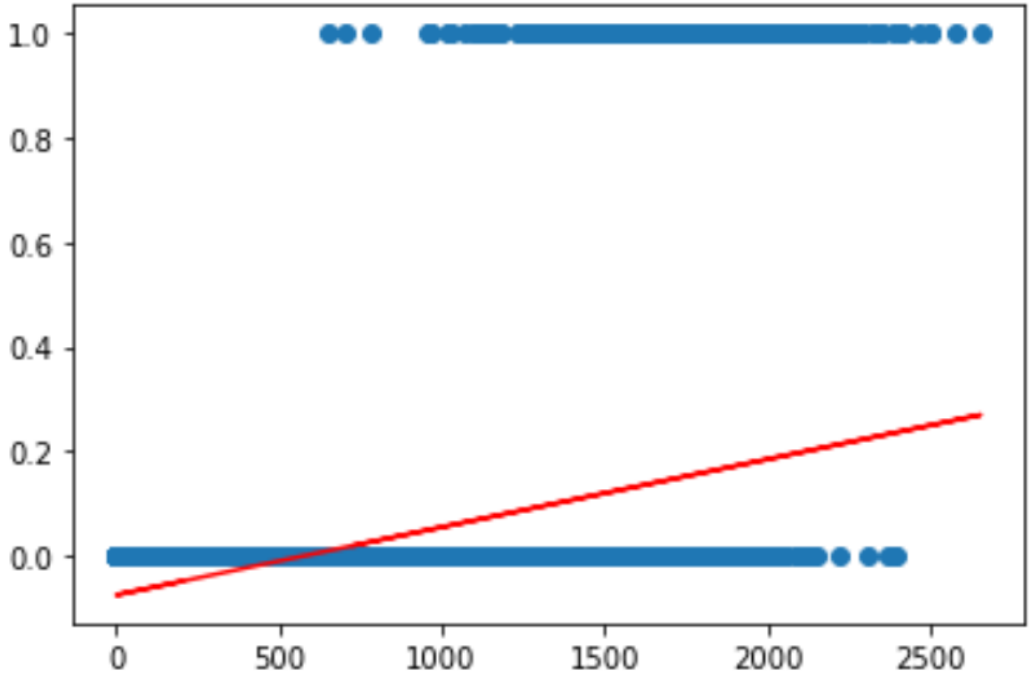
```
-0.07519195884353799
[0.00012987]
0.12258348714466394
```

Solve Q2, 3 and 4 of Part 2.

From linear to logistic regression (2/3)

```
Y_pred=lm.predict(X)
plt.scatter(X,Y)
plt.plot(X,Y_pred,color="red")
```

```
[<matplotlib.lines.Line2D at 0x2609ef15310>]
```




```
print(X.loc[1])
Y_pred[1]
```

```
balance    817.180407
Name: 1, dtype: float64

array([0.03093704])
```

Will this customer default?
Probably not…
Interpretation: Y_pred: probability of defaulting

```
print(X["balance"][9999])
Y_pred[9999]
```

```
200.9221826

array([-0.04909776])
```

From linear to logistic regression(3/3)

Conclusions:

- We want to interpret Y_pred as the **probability** that this particular person will default.
- How to go from the **probability** to a **class**? Goal is **classification.** Let's see later!

- Probabilities are **between 0 and 1.**
- **Linear regression** does **not work** to do this: get predictions that are smaller than 0, larger than 1.
- Need to fit a function to the data that only takes on values between 0 and 1.

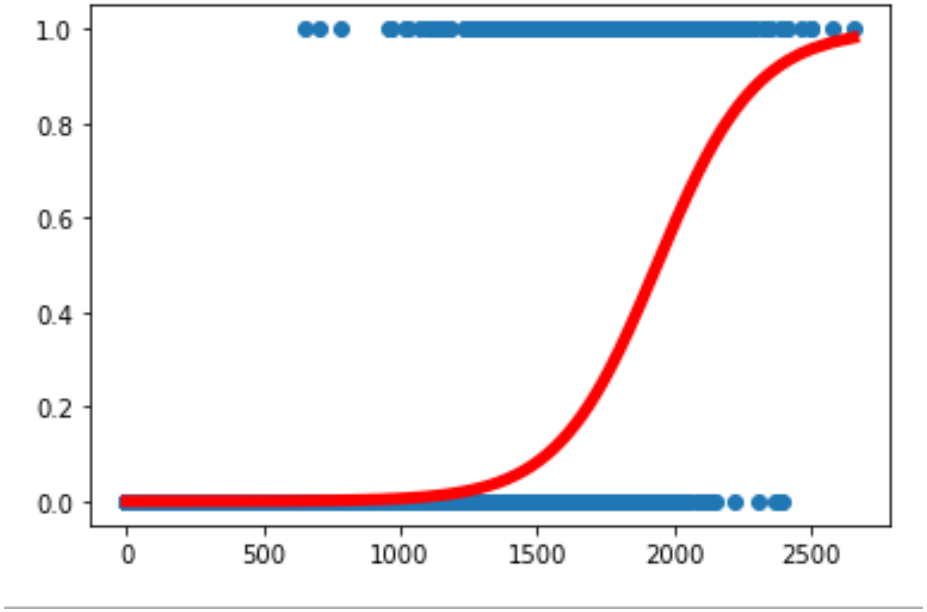
Logistic regression

INSEAD

The basics of logistic regression

Logistic regression (1/4)

What happens in logistic regression? We fit a **different curve.**



Input: datapoints (x_i, y_i)

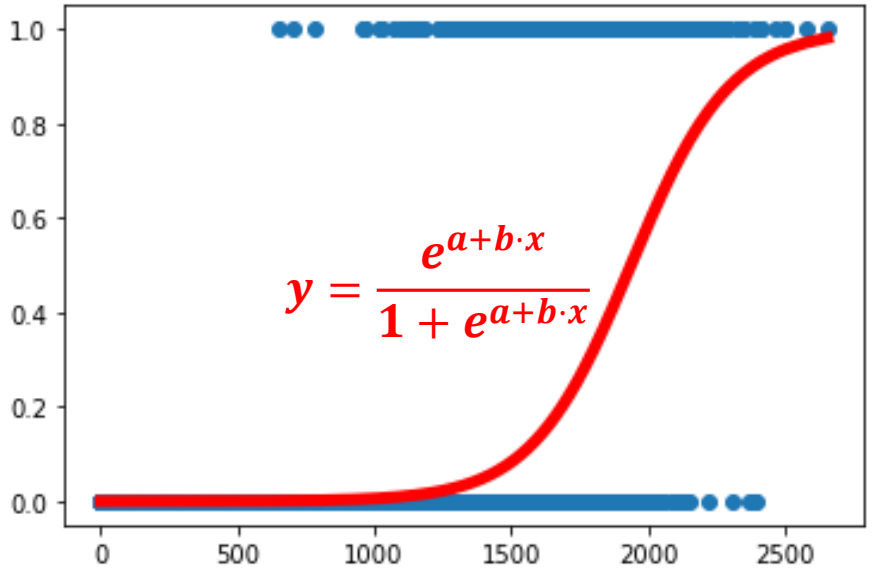
Here x_i is the balance; $y_i = 1$ (default) or 0 (doesn't default)

We have a 10000 datapoints.

Logistic regression (2/4)

Instead of $y = a + b \cdot x$ (line), we fit a different function:

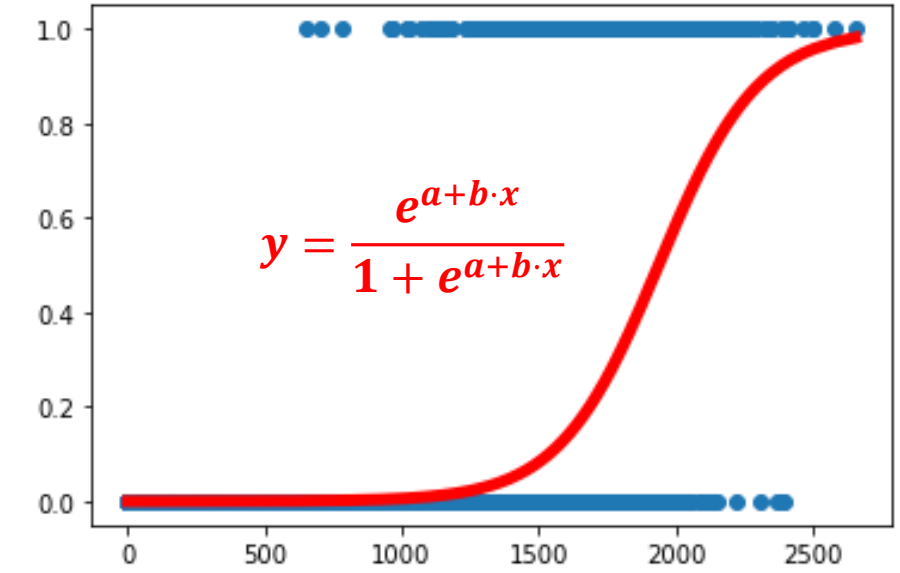
$$y = \frac{e^{a+b\cdot x}}{1 + e^{a+b\cdot x}}$$



Goal: Find numbers (a, b) such that

$\frac{e^{a+b\cdot x_i}}{1+e^{a+b\cdot x_i}}$ is as close as possible to y_i for all 10000 observations

Logistic regression (3/4)



Advantages:

$$y_{pred} = \frac{e^{a+b\cdot x}}{1+e^{a+b\cdot x}}$$

is a number between 0 and 1 (why?).

Output: The predicted value

$$y_{pred_i} = \frac{e^{a+b\cdot x_i}}{1+e^{a+b\cdot x_i}}$$

is always between 0 and 1 and can be interpreted as the **probability that observation x_i defaults (i.e., $P(y_i = 1)$).**

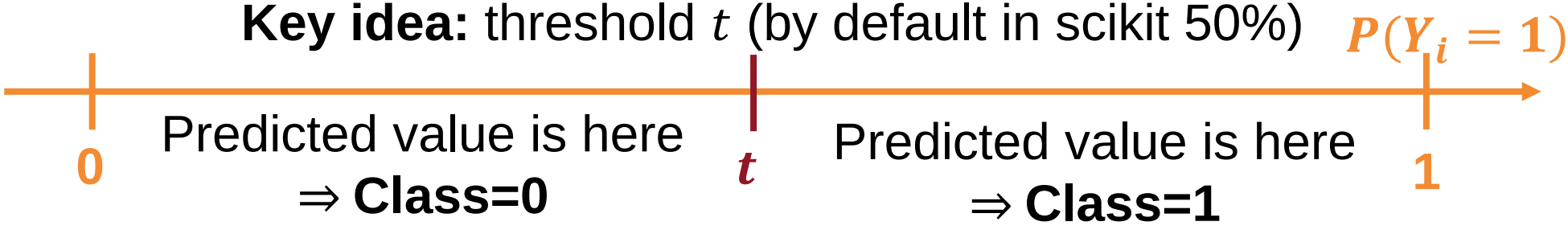
Logistic regression (4/4)

For each x_i,
we get a **probability (y_pred)** that the corresponding customer **will default**.

How to go from there to **classification**? i.e.,

How to **decide on labels** (defaults/doesn't default) to give to the customer?

Key idea: threshold t (by default in scikit 50%)



Logistic regression in Python

Your turn! Go through Part 3 of the notebook.

```
from sklearn.linear_model import LogisticRegression
```

```
#Logistic regression with scikit

X=Default[["balance"]]
Y=Default["default_Yes"]

logm = LogisticRegression().fit(X, Y) # Fit a logistic regression with vector Y as dependent and matrix X as independent

print("Intercept = ",logm.intercept_) # Print the resultant model intercept

print("Model coefficients = ", logm.coef_) # Print the resultant model coefficients (in order of variables in X)

print("R^2 =",logm.score(X,Y)) # Print the resultant model R-squared
```

```
logm.predict(X)
```

```
array([0, 0, 0, ..., 0, 0, 0], dtype=uint8)
```

Labels

```
logm.predict_proba(X)
```

```
array([[9.98694319e-01, 1.30568146e-03],
       [9.97887402e-01, 2.11259754e-03],
       [9.91405252e-01, 8.59474814e-03],
       ...,
       [9.97533484e-01, 2.46651596e-03],
       [8.83240365e-01, 1.16759635e-01],
       [9.99928552e-01, 7.14476480e-05]])
```

$P(Y = 0)$
(i.e., doesn't default)

Probabilities

$P(Y = 1)$
(i.e., defaults)

INSEAD

The logistic regression pipeline

Multivariate logistic regression (1/3)

- We only used the “balance” feature here to make things visual.
- **We should be using all our features here: this is multivariate logistic regression.**
- **All the issues** that can occur with linear regression can happen here too, e.g., **multicollinearity.**
- What is the **label here**? What are the **features**?

Multivariate logistic regression (2/3)

Answer questions 1) and 2) in the notebook in Part 4.

```
Y=Default["default_Yes"] #creating the d
X=Default.drop(columns=["default_Yes"])
```

```
X.corr()
```

	balance	income	student_Yes
balance	1.000000	-0.152243	0.203578
income	-0.152243	1.000000	-0.753985
student_Yes	0.203578	-0.753985	1.000000

```
X=X.drop(columns=["income"])
X
```

Multivariate logistic regression (3/3)

We now practice the **supervised machine learning pipeline.**

- Split the data into train and test. Why is there no validation?
- **Complete questions 3 and 4 in Part 4.**

```
trainX, testX, trainY, testY = train_test_split(X, Y, test_size=0.25)
```

```
#Logistic regression with scikit

logm = LogisticRegression().fit(trainX, trainY) # Fi

print("Intercept = ",logm.intercept_) # Print the re

print("Model coefficients = ", logm.coef_) # Print t
```

```
Intercept =  [-10.7305728]
Model coefficients =  [[ 0.00569611 -0.73717829]]
```

Metrics for classification (1/4)

How should we **use our testing set?**

- For **regression**, we used the **RMSE** to compare the predicted values to the true values.
- For **classification**, our true values are “default/did not default”. How do we compare these to our predicted classes?

Confusion matrix

	Predicted class=0 (i.e. predicted to not default)	**Predicted class=1 (i.e. predicted to default)**
Actual class =0 (i.e. actual non-defaults)	True Negatives (TN)	False Positives (FP)
Actual class =1 (i.e. actual defaults)	False Negatives (FN)	True Positives (TP)

Metrics for classification (2/4)

	Predicted class=0 (i.e. predicted to not default)	**Predicted class=1 (i.e. predicted to default)**
Actual class =0 (i.e. actual non-defaults)	True Negatives (TN)	False Positives (FP)
Actual class =1 (i.e. actual defaults)	False Negatives (FN)	True Positives (TP)

Specificity $= \frac{TN}{FP+TN}$
(proportion of actual negatives correctly identified)

Sensitivity $= \frac{TP}{TP+FN}$
(proportion of actual positives correctly identified)

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

Metrics for classification (3/4)

Which error is it most important for the bank to minimize? FP or FN?

	Predicted class=0 (i.e. predicted to not default)	**Predicted class=1 (i.e. predicted to default)**
Actual class =0 (i.e. actual non-defaults)	True Negatives (TN)	False Positives (FP)
Actual class =1 (i.e. actual defaults)	False Negatives (FN)	True Positives (TP)

Predicted to default but did not default

Predicted to not default but did default

⇒ Probably better to have low FN (bad surprises) than low FP.

Complete Q4 of P4

Metrics for classification (4/4)

```
Y_pred=logm.predict(testX)
from sklearn.metrics import confusion_matrix
confusion_matrix(testY,Y_pred)
```

Compare the true values on the testing set to the predicted classes. **Order is very important here!!**

What do you think of the confusion matrix?

```
array([[2411,   13],
       [  49,   27]], dtype=int64)
```

We would like to have lower FN maybe, even if it possibly comes with higher FPs… How to do this?

Change the threshold! (Next lecture)

Wrap-up & Next time

Today, we:

- Learnt the difference between **classification and regression**
- **Understood logistic regression**
- **Saw** the **differences** between logistic and linear regression.
- Saw how to use a **confusion matrix**!

Next time:

- Do quiz before next lecture
- More logistic regression
- **No homework… so you can go over today's lecture carefully and be ready for next lecture.**

The Business School
for the World®

EUROPE | ASIA | MIDDLE EAST | NORTH AMERICA