Week 3 index 1: Introduction to Classification

In machine learning classification is a supervised learning approach which can be thought of as a means of categorizing or classifying some unknown items into a discrete set of classes.

Classification attempts to learn the relationship between a set of feature variables and a target variable of interest.

The target attribute in classification is a categorical variable with discrete values.

What is classification?

- A supervised learning approach
- Categorizing some unknown items into a discrete set of categories or "classes"
- The target attribute is a categorical variable

So, how does classification and classifiers work?

Given a set of training data points along with the target labels, classification determines the class label for an unlabeled test case.

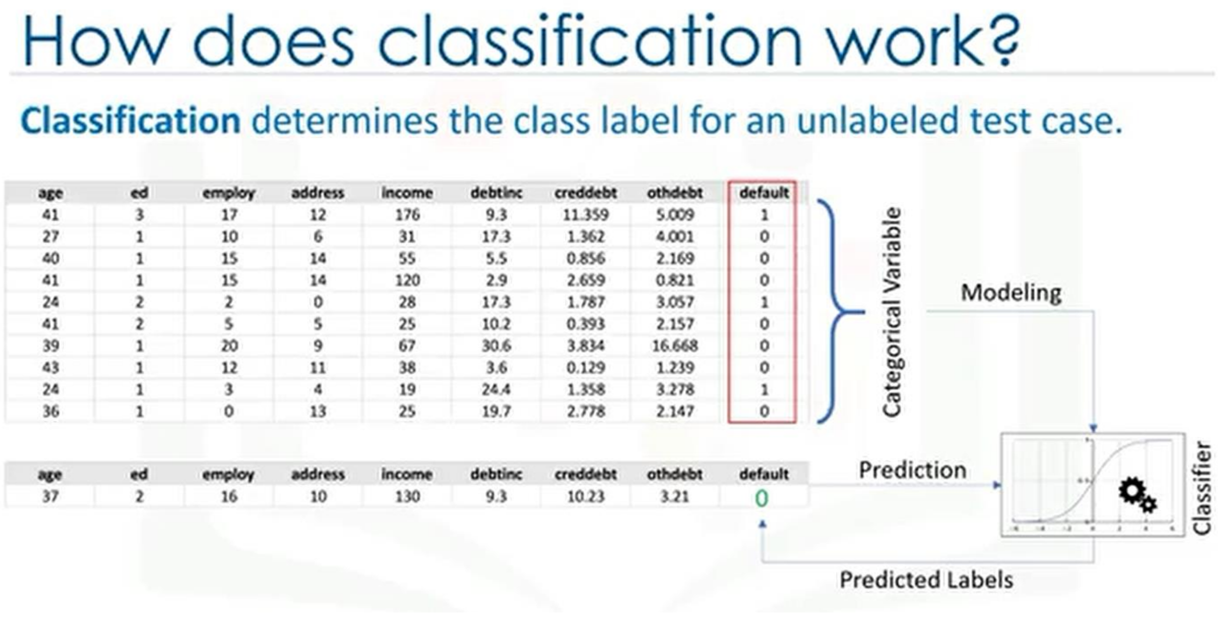
Let's explain this with an example. A good sample of classification is the loan default prediction.

Suppose a bank is concerned about the potential for loans not to be repaid?

If previous loan default data can be used to predict which customers are likely to have problems repaying loans, these bad risk customers can either have their loan application declined or offered alternative products.

The goal of a loan default predictor is to use existing loan default data which has information about the customers such as age, income, education et cetera, to build a classifier, pass a new customer or potential future default to the model, and then label it, i.e the data points as defaulter or not defaulter. Or for example zero or one.

This is how a classifier predicts an unlabeled test case.



How does classification work?

Classification determines the class label for an unlabeled test case.

age	ed	employ	address	income	debtinc	creddebt	othdebt	default
41	3	17	12	176	9.3	11.359	5.009	1
27	1	10	6	31	17.3	1.362	4.001	0
40	1	15	14	55	5.5	0.856	2.169	0
41	1	15	14	120	2.9	2.659	0.821	0
24	2	2	0	28	17.3	1.787	3.057	1
41	2	5	5	25	10.2	0.393	2.157	0
39	1	20	9	67	30.6	3.834	16.668	0
43	1	12	11	38	3.6	0.129	1.239	0
24	1	3	4	19	24.4	1.358	3.278	1
36	1	0	13	25	19.7	2.778	2.147	0

age	ed	employ	address	income	debtinc	creddebt	othdebt	default
37	2	16	10	130	9.3	10.23	3.21	0

Note: Please notice that this specific example was about a binary classifier with two values. We can also build classifier models for both binary classification and multi-class classification.

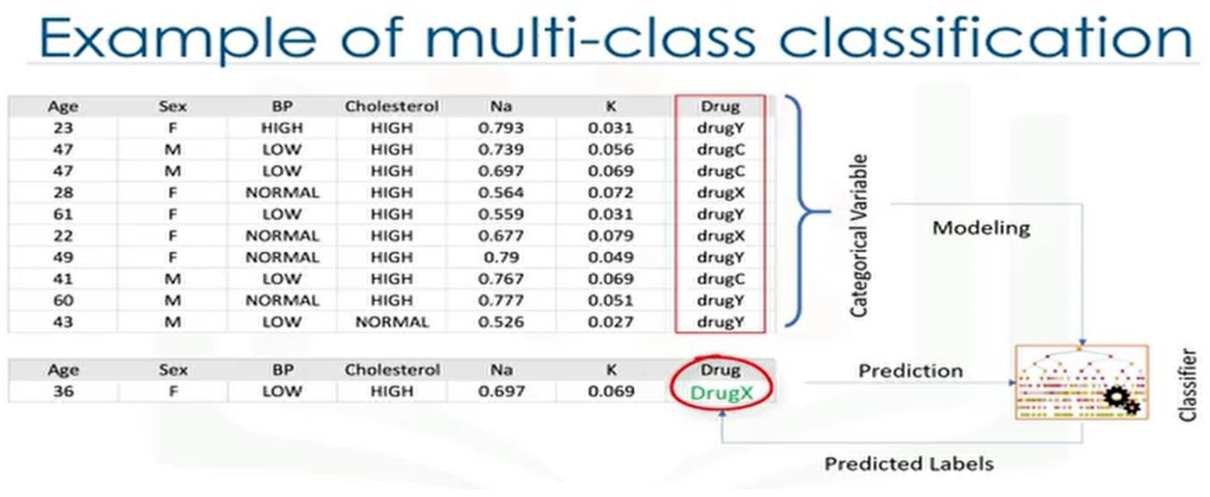
Q: What is Multi-class classifier?

A classifier that can predict a field with multiple discrete values, such as Drug x, Drug y, Drug z

For example, imagine that you've collected data about a set of patients, all of whom suffered from the same illness.

During their course of treatment, each patient responded to one of three medications. You can use this labeled dataset with a classification algorithm to build a classification model.

Then you can use it to find out which drug might be appropriate for a future patient with the same illness. As you can see, it is a sample of multi-class classification.



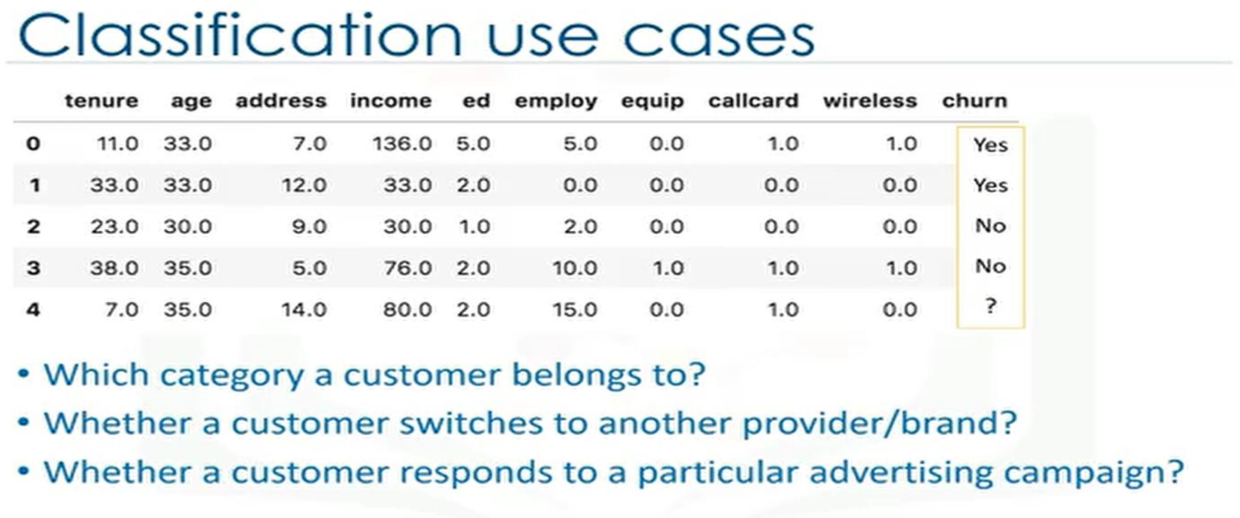
Example of multi-class classification

Age	Sex	BP	Cholesterol	Na	K	Drug
23	F	HIGH	HIGH	0.793	0.031	drugY
47	M	LOW	HIGH	0.739	0.056	drugC
47	M	LOW	HIGH	0.697	0.069	drugC
28	F	NORMAL	HIGH	0.564	0.072	drugX
61	F	LOW	HIGH	0.559	0.031	drugY
22	F	NORMAL	HIGH	0.677	0.079	drugX
49	F	NORMAL	HIGH	0.79	0.049	drugY
41	M	LOW	HIGH	0.767	0.069	drugC
60	M	NORMAL	HIGH	0.777	0.051	drugY
43	M	LOW	NORMAL	0.526	0.027	drugY

Age	Sex	BP	Cholesterol	Na	K	Drug
36	F	LOW	HIGH	0.697	0.069	DrugX

Classification has different business use cases as well.

For example, to predict the category to which a customer belongs, for churn detection where we predict whether a customer switches to another provider or brand, or to predict whether or not a customer responds to a particular advertising campaign.



Classification use cases

	tenure	age	address	income	ed	employ	equip	callcard	wireless	churn
0	11.0	33.0	7.0	136.0	5.0	5.0	0.0	1.0	1.0	Yes
1	33.0	33.0	12.0	33.0	2.0	0.0	0.0	0.0	0.0	Yes
2	23.0	30.0	9.0	30.0	1.0	2.0	0.0	0.0	0.0	No
3	38.0	35.0	5.0	76.0	2.0	10.0	1.0	1.0	1.0	No
4	7.0	35.0	14.0	80.0	2.0	15.0	0.0	1.0	0.0	?

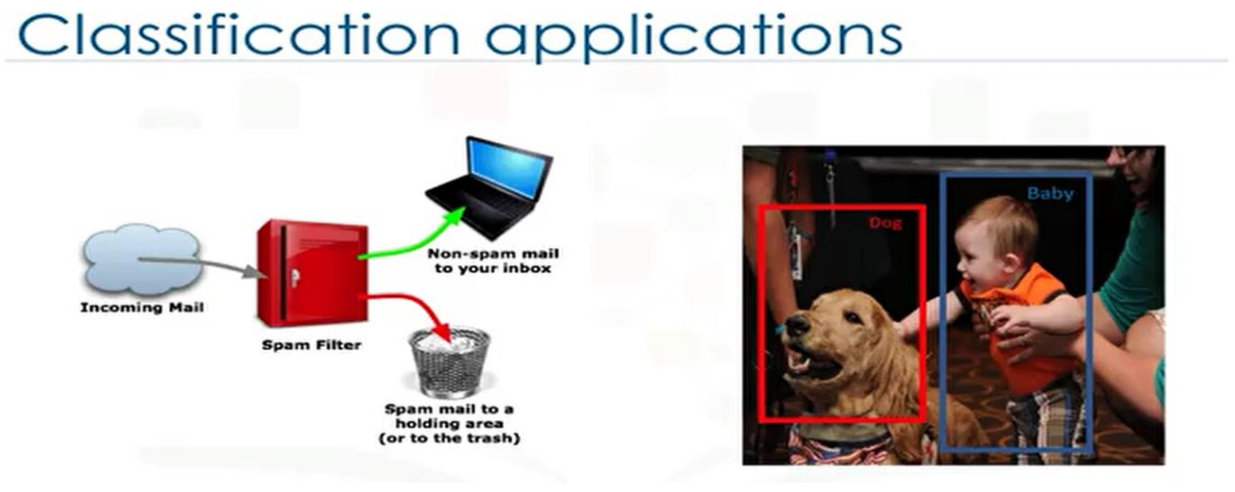
- Which category a customer belongs to?
- Whether a customer switches to another provider/brand?
- Whether a customer responds to a particular advertising campaign?

Data classification has several applications in a wide variety of industries. Essentially, many problems can be expressed as associations between feature and target variables, especially when labelled data is available. This provides a broad range of applicability for classification.

For example, classification can be used for

- email filtering,
- speech recognition,

- handwriting recognition,
- biometric identification,
- document classification and much more.



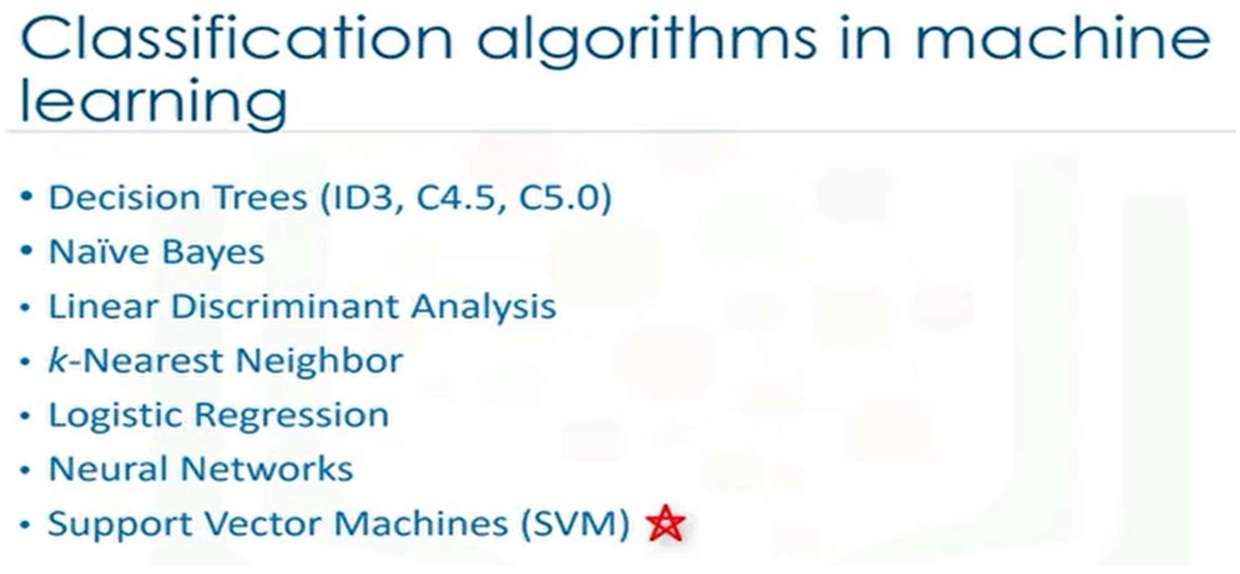
Classification applications

Incoming Mail
Spam Filter
Non-spam mail to your inbox
Spam mail to a holding area (or to the trash)
Dog
Baby

we have the types of classification algorithms and machine learning. They include:

- ✓ decision trees,
- ✓ naive bayes,
- ✓ linear discriminant analysis,
- ✓ k-nearest neighbor,
- ✓ logistic regression,
- ✓ neural networks, and
- ✓ support vector machines.

There are many types of classification algorithms.



Classification algorithms in machine learning

- Decision Trees (ID3, C4.5, C5.0)
- Naïve Bayes
- Linear Discriminant Analysis
- *k*-Nearest Neighbor
- Logistic Regression
- Neural Networks
- Support Vector Machines (SVM) ☆

Week 3 index 2: K-Nearest Neighbors

In this video, we'll be covering the K-Nearest Neighbors algorithm. So, let's get started. Imagine that a telecommunications provider has segmented his customer base by service usage patterns, categorizing the customers into four groups.

If demographic data can be used to predict group membership, the company can customize offers for individual perspective customers.

This is a classification problem. That is, given the dataset with predefined labels, we need to build a model to be used to predict the class of a new or unknown case.

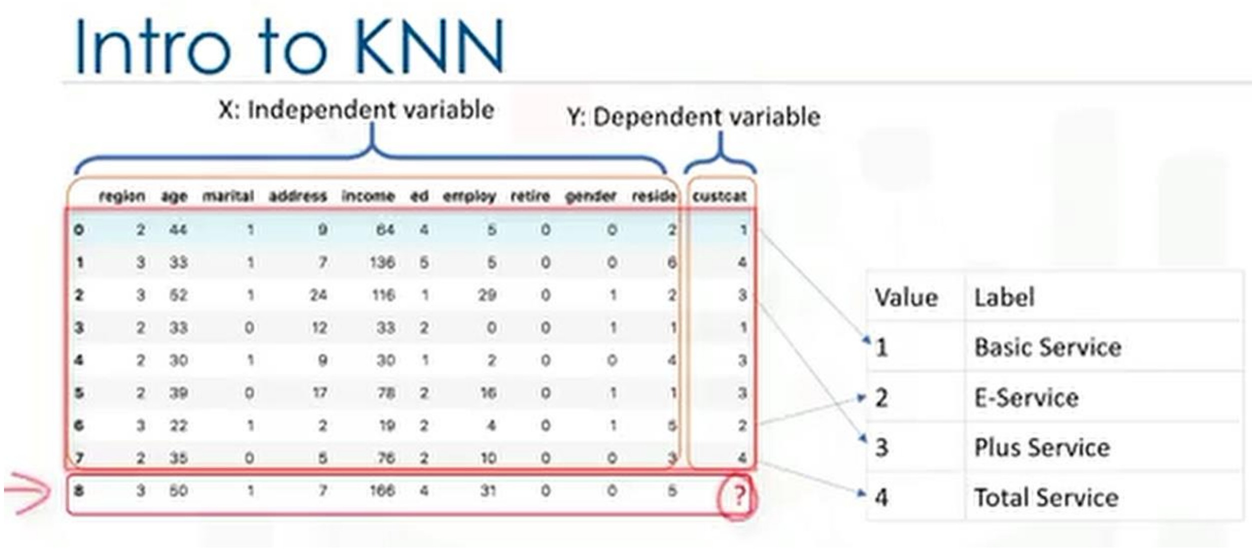
The example focuses on using demographic data, such as region, age, and marital status to predict usage patterns.

# The target field called **custcat** has four possible values that correspond to the four customer groups as follows:

- Basic Service,
- E Service,
- Plus Service,
- Total Service.

Our objective is to build a classifier.

For example, using the row zero to seven to predict the class of row eight. We will use a specific type of classification called K-Nearest Neighbor.



Intro to KNN

X: Independent variable
Y: Dependent variable

	region	age	marital	address	income	ed	employ	retire	gender	reside	custcat
0	2	44	1	9	64	4	5	0	0	2	1
1	3	33	1	7	136	5	5	0	0	6	4
2	3	52	1	24	116	1	29	0	1	2	3
3	2	33	0	12	33	2	0	0	1	1	1
4	2	30	1	9	30	1	2	0	0	4	3
5	2	39	0	17	78	2	16	0	1	1	3
6	3	22	1	2	19	2	4	0	1	5	2
7	2	35	0	5	76	2	10	0	0	3	4
8	3	50	1	7	166	4	31	0	0	5	?

Value	Label
1	Basic Service
2	E-Service
3	Plus Service
4	Total Service

Just for sake of demonstration, let's use only two fields as predictors specifically, age and income, and then plot the customers based on their group membership. Now, let's say that we have a new customer.

For example, record number eight, with a known age and income.

- ✓ **Q:** How can we find the class of this customer?
- ✓ **Q:** Can we find one of the closest cases and assign the same class label to our new customer?
- ✓ **Q:** Can we also say that the class of our new customer is most probably group four i.e Total Service, because its nearest neighbor is also of class four?

Yes, we can. In fact, it is the first nearest neighbor. Now, the question is to

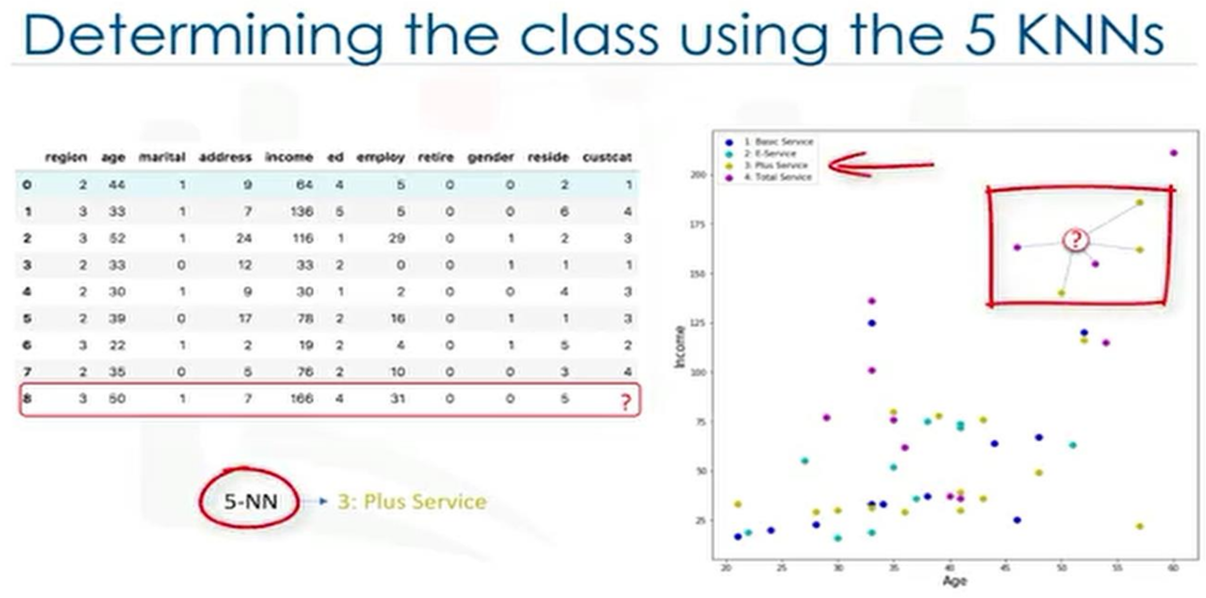
Q: what extent can we trust our judgment which is based on the first nearest neighbor?

It might be a poor judgment especially if the first nearest neighbor is a very specific case or an outlier, correct?

Now, let's look at our scatter plot again. Rather than choose the first nearest neighbor,

Q: what if we chose the five nearest neighbors and did a majority vote among them to define the class of our new customer?

In this case, we'd see that three out of five nearest neighbors tell us to go for class three, which is Plus Service.



Determining the class using the 5 KNNs

	region	age	marital	address	income	ed	employ	retire	gender	reside	custcat
0	2	44	1	9	64	4	5	0	0	2	1
1	3	33	1	7	136	5	5	0	0	6	4
2	3	52	1	24	116	1	29	0	1	2	3
3	2	33	0	12	33	2	0	0	1	1	1
4	2	30	1	9	30	1	2	0	0	4	3
5	2	39	0	17	78	2	16	0	1	1	3
6	3	22	1	2	19	2	4	0	1	5	2
7	2	35	0	5	76	2	10	0	0	3	4
8	3	50	1	7	166	4	31	0	0	5	?

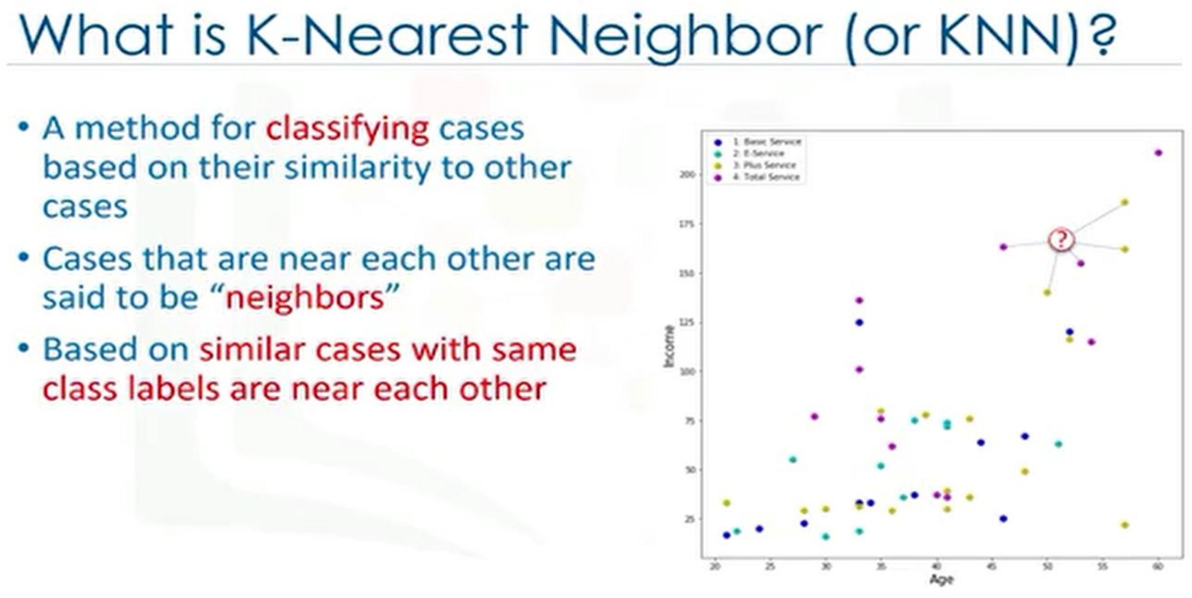
5-NN → 3: Plus Service

Doesn't this make more sense? Yes. In fact, it does. In this case, the value of K in the K-Nearest Neighbors algorithm is five.

This example highlights the intuition behind the K-Nearest Neighbors algorithm. Now, let's define the K Nearest Neighbors.

K-Nearest Neighbors: The K-Nearest Neighbors algorithm is a classification algorithm that takes a bunch of labeled points and uses them to learn how to label other points.

- ✓ This algorithm classifies cases based on their similarity to other cases.
- ✓ In K-Nearest Neighbors, data points that are near each other are said to be neighbors.
- ✓ K-Nearest Neighbors is based on this paradigm. Similar cases with the same class labels are near each other.



What is K-Nearest Neighbor (or KNN)?

- A method for classifying cases based on their similarity to other cases
- Cases that are near each other are said to be "neighbors"
- Based on similar cases with same class labels are near each other

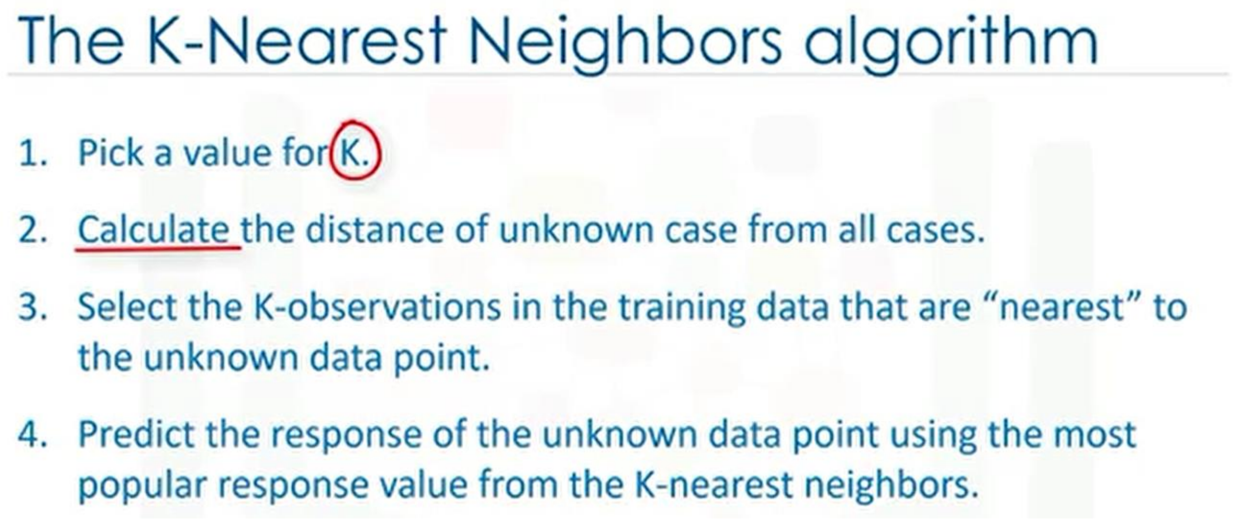
Thus, the distance between two cases is a measure of their dissimilarity.

There are different ways to calculate the similarity or conversely, the distance or dissimilarity of two data points.

For example, this can be done using **Euclidean distance**.

Now, let's see how the K-Nearest Neighbors algorithm actually works. In a classification problem, the K-Nearest Neighbors algorithm works as follows.

- One, pick a value for K.
- Two, calculate the distance from the new case hold out from each of the cases in the dataset.
- Three, search for the K-observations in the training data that are nearest to the measurements of the unknown data point.
- And four, predict the response of the unknown data point using the most popular response value from the K-Nearest Neighbors.



The K-Nearest Neighbors algorithm

1. Pick a value for K.
2. Calculate the distance of unknown case from all cases.
3. Select the K-observations in the training data that are "nearest" to the unknown data point.
4. Predict the response of the unknown data point using the most popular response value from the K-nearest neighbors.

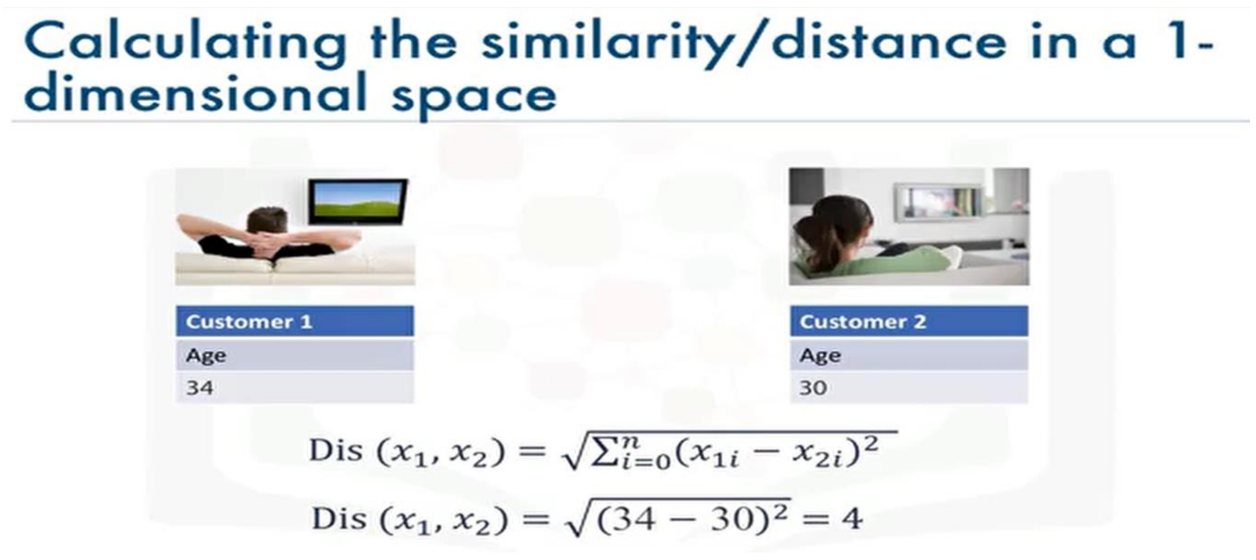
There are two parts in this algorithm that might be a bit confusing.

- ✓ First, how to select the correct K and
- ✓ second, how to compute the similarity between cases, **for example**, among customers.

Let's first start with the second concern.

That is, how can we calculate the similarity between two data points?

Assume that we have two customers, customer one and customer two, and for a moment, assume that these two customers have only one feature, Age. We can easily use a specific type of Makowski distance to calculate the distance of these two customers, it is indeed the Euclidean distance. Distance of X_1 from X_2 is:



Calculating the similarity/distance in a 1-dimensional space

Customer 1
Age
34

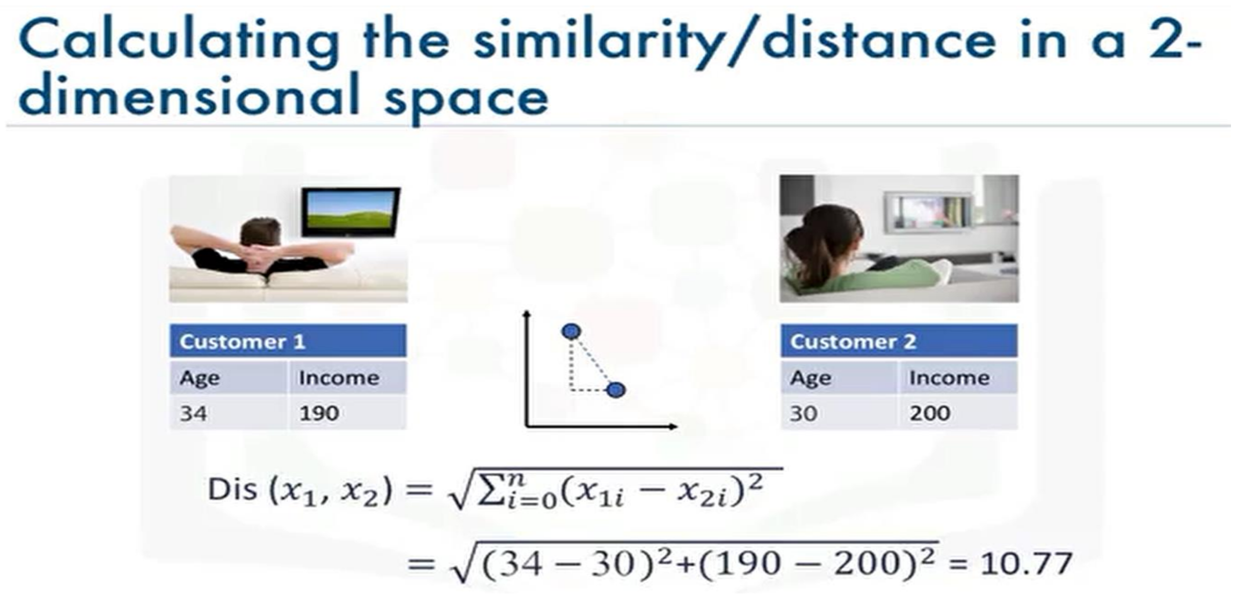
Customer 2
Age
30

$$\text{Dis}\,(x_1, x_2) = \sqrt{\textstyle\sum_{i=0}^{n}(x_{1i} - x_{2i})^2}$$

$$\text{Dis}\,(x_1, x_2) = \sqrt{(34-30)^2} = 4$$

What about if we have more than one feature?

For example, age and income. If we have income and age for each customer, we can still use the same formula but this time, we're using it in a two-dimensional space.



Calculating the similarity/distance in a 2-dimensional space

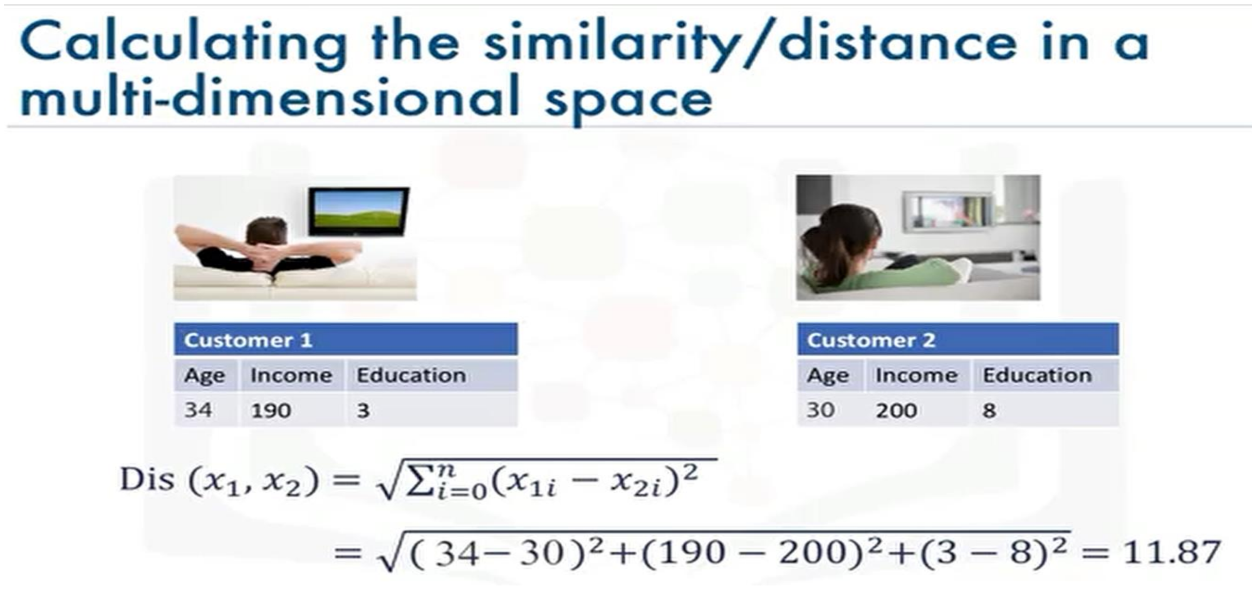
Customer 1	
Age	Income
34	190

Customer 2	
Age	Income
30	200

$$\text{Dis}\,(x_1, x_2) = \sqrt{\textstyle\sum_{i=0}^{n}(x_{1i} - x_{2i})^2}$$

$$= \sqrt{(34-30)^2 + (190-200)^2} = 10.77$$

We can also use the same distance matrix for multidimensional vectors.


Calculating the similarity/distance in a multi-dimensional space

Customer 1		
Age	Income	Education
34	190	3

Customer 2		
Age	Income	Education
30	200	8

$$\text{Dis}\ (x_1, x_2) = \sqrt{\sum_{i=0}^{n}(x_{1i} - x_{2i})^2}$$
$$= \sqrt{(34-30)^2+(190-200)^2+(3-8)^2} = 11.87$$

Of course, we have to normalize our feature set to get the accurate dissimilarity measure.

There are other dissimilarity measures as well that can be used for this purpose but as mentioned, it is highly dependent on datatype and also the domain that classification is done for it.

As mentioned, K and K-Nearest Neighbors is the number of nearest neighbors to examine. It is supposed to be specified by the user.

Q: So, how do we choose the right K?

Assume that we want to find the class of the customer noted as question mark on the chart.

Q: What happens if we choose a very low value of K?

Let's say, K=1. The first nearest point would be blue, which is class one.

This would be a bad prediction, since more of the points around it are magenta or class four.

Note: In fact, since its nearest neighbor is blue we can say that we capture the noise in the data or we chose one of the points that was an anomaly in the data.

Note: A low value of K causes a highly complex model as well, which might result in overfitting of the model. It means the prediction process is not generalized enough to be used for out-of-sample cases.

Note: Out-of-sample data is data that is outside of the data set used to train the model. In other words, it cannot be trusted to be used for prediction of unknown samples.

Note: It's important to remember that overfitting is bad, as we want a general model that works for any data, not just the data used for training.

Now, on the opposite side of the spectrum, if we choose a very high value of K such as K=20, then the model becomes overly generalized.

To find the best value for K

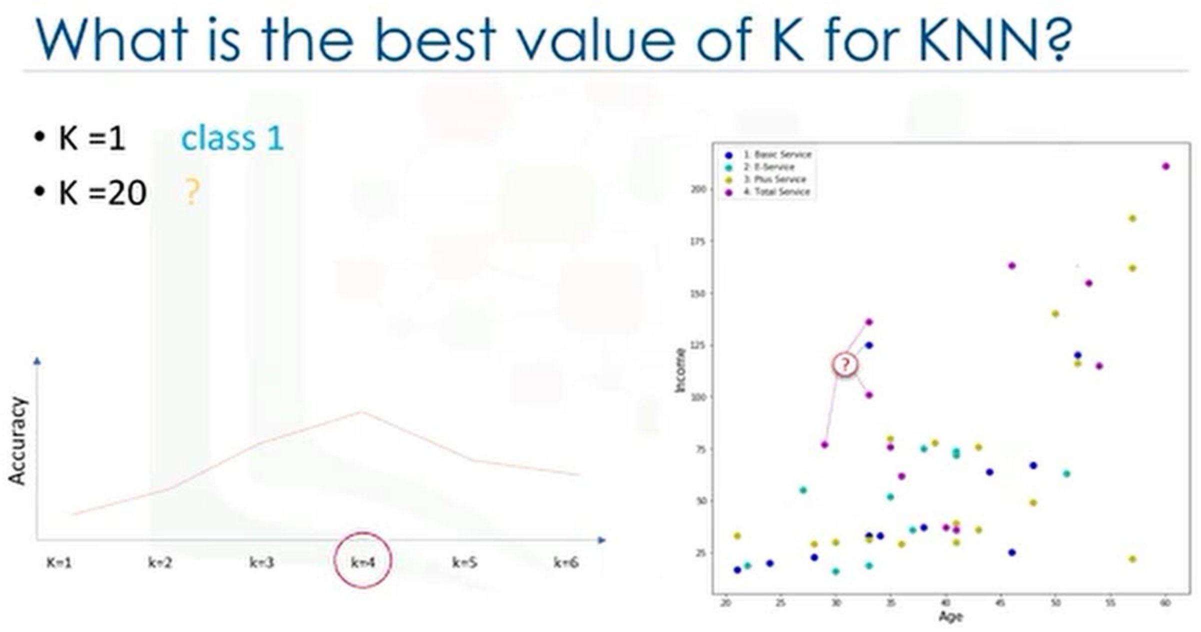
Q: So, how can we find the best value for K?

The general solution is to reserve a part of your data for testing the accuracy of the model.

Once you've done so, choose K=1 and then use the training part for modeling and calculate the accuracy of prediction using all samples in your test set.

Repeat this process increasing the K and see which K is best for your model.

For example, in our case, K equals four will give us the best accuracy.



What is the best value of K for KNN?

- K =1 class 1
- K =20 ?

Note: Nearest neighbor analysis can also be used to compute values for a continuous target.

In this situation, the average or median target value of the nearest neighbors is used to obtain the predicted value for the new case.

For example, assume that you are predicting the price of a home based on its feature set, such as

- number of rooms,
- square footage,
- the year it was built, and so on.

You can easily find the three nearest neighbor houses of course not only based on distance but also based on all the attributes and then predict the price of the house as the medium of neighbors.

Computing continuous targets using KNN

- KNN can also be used for regression

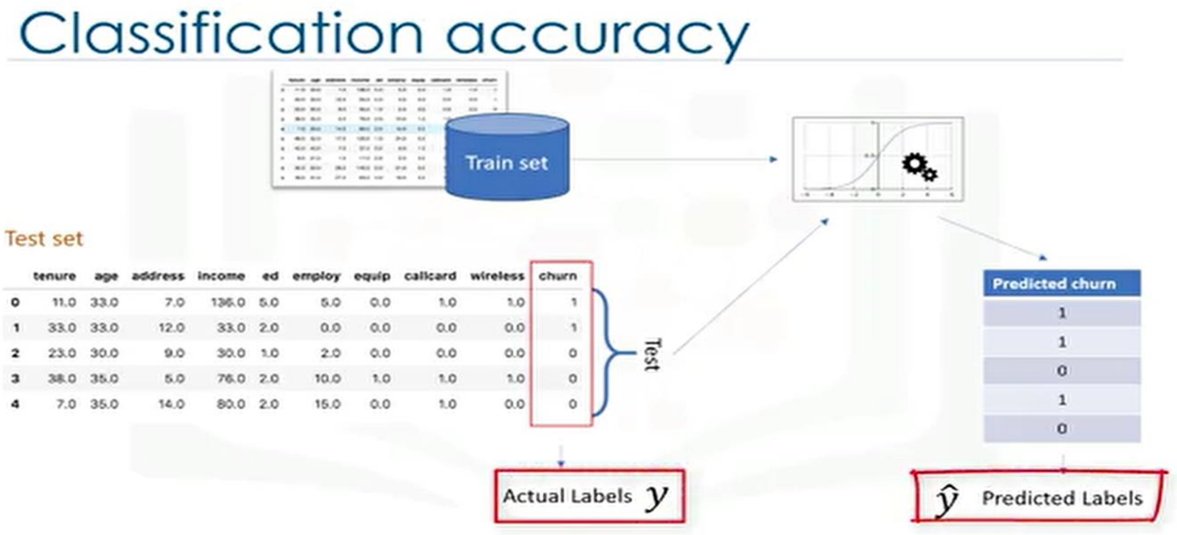
Week 3 (index 3): Evaluation Metrics in Classification

In this video, we'll be covering evaluation metrics for classifiers. So, let's get started. Evaluation metrics explain the performance of a model. Let's talk more about the model evaluation metrics that are used for classification.

Imagine that we have an historical dataset which shows the customer churn for a telecommunication company. We have trained the model, and now we want to calculate its accuracy using the test set. We pass the test set to our model, and we find the predicted labels.

Q: Now the question is, "How accurate is this model?"

Basically, we compare the actual values in the test set with the values predicted by the model, to calculate the accuracy of the model.



	tenure	age	address	income	ed	employ	equip	callcard	wireless	churn
0	11.0	33.0	7.0	136.0	5.0	5.0	0.0	1.0	1.0	1
1	33.0	33.0	12.0	33.0	2.0	0.0	0.0	0.0	0.0	1
2	23.0	30.0	9.0	30.0	1.0	2.0	0.0	0.0	0.0	0
3	38.0	35.0	5.0	76.0	2.0	10.0	1.0	1.0	1.0	0
4	7.0	35.0	14.0	80.0	2.0	15.0	0.0	1.0	0.0	0

Predicted churn
1
1
0
1
0

Note: Evaluation metrics provide a key role in the development of a model, as they provide insight to areas that might require improvement.

There are different model evaluation metrics but we just talk about three of them here, specifically:

- Jaccard index,
- F1-score, and
- Log Loss.

Jaccard index

Let’s first look at one of the simplest accuracy measurements, the Jaccard index -- also known as the Jaccard similarity coefficient.

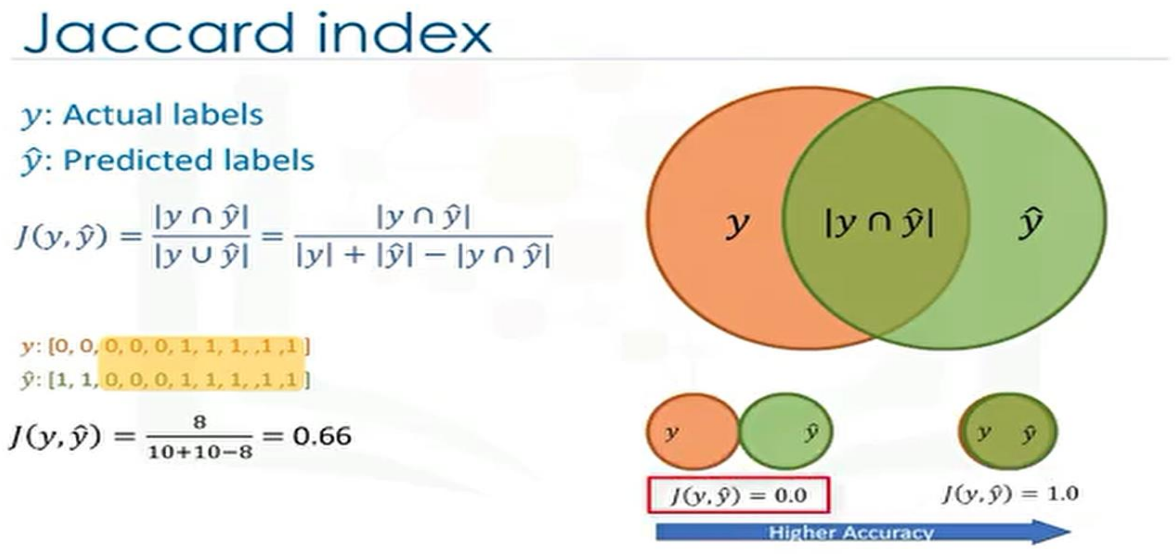
Let’s say

- ✓ y shows the true labels of the churn dataset. And
- ✓ $\hat{y}$ shows the predicted values by our classifier.

Then we can define Jaccard as the size of the intersection divided by the size of the union of two label sets.

For example, for a test set of size 10, with 8 correct predictions, or 8 intersections, the accuracy by the Jaccard index would be 0.66.

Note: If the entire set of predicted labels for a sample strictly matches with the true set of labels, then the subset accuracy is 1.0; otherwise, it is 0.0.


Jaccard index
y: Actual labels
ŷ: Predicted labels
J(y, ŷ) = |y ∩ ŷ| / |y ∪ ŷ| = |y ∩ ŷ| / (|y| + |ŷ| − |y ∩ ŷ|)
y: [0, 0, 0, 0, 0, 1, 1, 1, ,1 ,1]
ŷ: [1, 1, 0, 0, 0, 1, 1, 1, ,1 ,1]
J(y, ŷ) = 8 / (10+10−8) = 0.66
y
|y ∩ ŷ|
ŷ
y
ŷ
y ŷ
J(y, ŷ) = 0.0
J(y, ŷ) = 1.0
Higher Accuracy

F1-score

Another way of looking at accuracy of classifiers is to look at a confusion matrix.

For example, let’s assume that our test set has only 40 rows.

Note: This matrix shows the corrected and wrong predictions, in comparison with the actual labels.

Each confusion matrix

- ✓ row shows the Actual/True labels in the test set, and
- ✓ the columns show the predicted labels by classifier.

let's Look at the first row. The first row is for customers whose actual churn value in the test set is 1.

- ❖ As you can calculate, out of 40 customers, the churn value of 15 of them is 1.
- ❖ And out of these 15, the classifier correctly predicted 6 of them as 1, and 9 of them as 0.

This means that for 6 customers, the actual churn value was 1, in the test set, and the classifier also correctly predicted those as 1.

However, while the actual label of 9 customers was 1, the classifier predicted those as 0, which is not very good.

We can consider this as an error of the model for the first row.

Q: What about the customers with a churn value 0?

Let's look at the second row. It looks like there were 25 customers whose churn value was 0. The classifier correctly predicted 24 of them as 0, and one of them wrongly predicted as 1.

So, it has done a good job in predicting the customers with a churn value of 0.

Note: A good thing about the confusion matrix is that it shows the model's ability to correctly predict or separate the classes.

In the specific case of a binary classifier, such as this example,

we can interpret these numbers as the count of true positives, false negatives, true negatives, and false positives.

Based on the count of each section, we can calculate the precision and recall of each label.

Precision is a measure of the accuracy, provided that a class label has been predicted. It is defined by:

precision = True Positive / (True Positive + False Positive).

And Recall is the true positive rate. It is defined as:

Recall = True Positive / (True Positive + False Negative).

So, we can calculate the precision and recall of each class.

Now we're in the position to calculate the F1 scores for each label, based on the precision and recall of that label.

Note: The F1 score is the harmonic average of the precision and recall, where an F1 score reaches its best value at 1 (which represents perfect precision and recall) and its worst at 0. It is a good way to show that a classifier has a good value for both recall and precision.

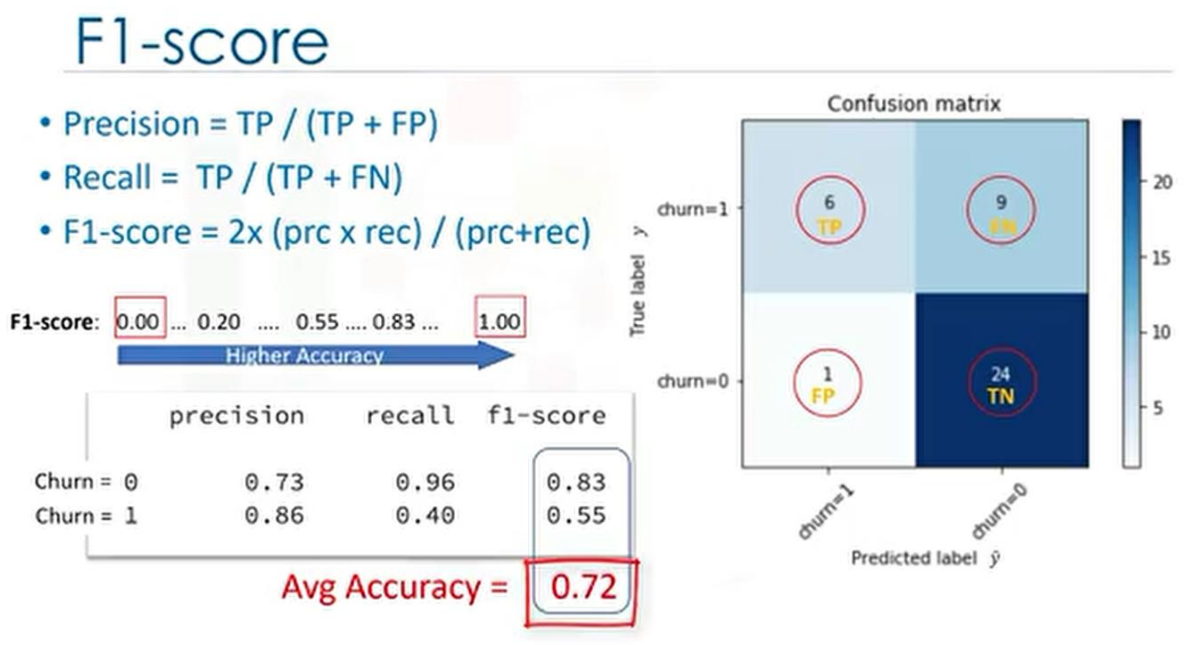
It is defined using the F1-score equation.

For example,

the F1-score for class 0 (i.e. churn=0), is 0.83, and

the F1-score for class 1 (i.e. churn=1), is 0.55.

And finally, we can tell the average accuracy for this classifier is the average of the F1-score for both labels, which is 0.72 in our case.



F1-score

- Precision = TP / (TP + FP)
- Recall = TP / (TP + FN)
- F1-score = 2x (prc x rec) / (prc+rec)

F1-score: $\boxed{0.00}$ ... 0.20 .... 0.55 .... 0.83 ... $\boxed{1.00}$

	precision	recall	f1-score
Churn = 0	0.73	0.96	0.83
Churn = 1	0.86	0.40	0.55

Avg Accuracy = $\boxed{0.72}$

Note: Please notice that both Jaccard and F1-score can be used for multi-class classifiers as well, which is out of scope for this course.

Log Loss

Now let's look at another accuracy metric for classifiers.

Note: Sometimes, the output of a classifier is the probability of a class label, instead of the label.

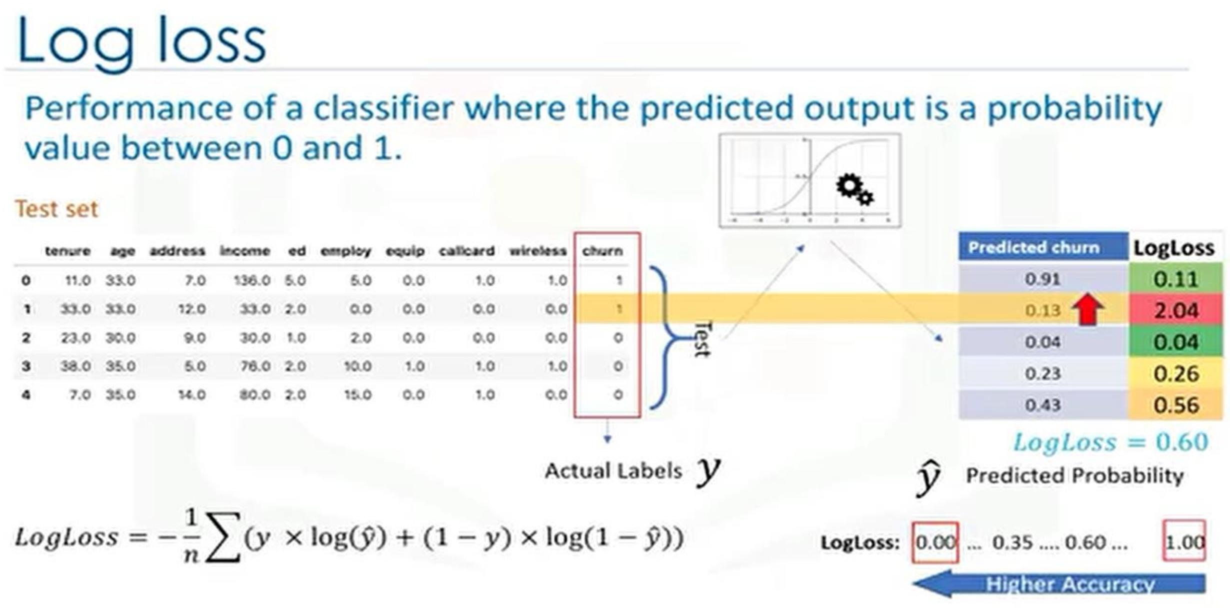
For example, in logistic regression, the output can be the probability of customer churn, i.e., yes (or equals to 1). This probability is a value between 0 and 1.

Logarithmic loss: Logarithmic loss (also known as Log loss) measures the performance of a classifier where the predicted output is a probability value between 0 and 1.

So, **for example,** predicting a probability of 0.13 when the actual label is 1, would be bad and would result in a high log loss.

We can calculate the log loss for each row using the log loss equation, which measures how far each prediction is, from the actual label. Then, we calculate the average log loss across all rows of the test set.

Note: It is obvious that ideal classifiers have progressively smaller values of log loss. So, the classifier with lower log loss has better accuracy.



Log loss

Performance of a classifier where the predicted output is a probability value between 0 and 1.

Test set

	tenure	age	address	income	ed	employ	equip	callcard	wireless	churn
0	11.0	33.0	7.0	136.0	5.0	5.0	0.0	1.0	1.0	1
1	33.0	33.0	12.0	33.0	2.0	0.0	0.0	0.0	0.0	1
2	23.0	30.0	9.0	30.0	1.0	2.0	0.0	0.0	0.0	0
3	38.0	35.0	5.0	76.0	2.0	10.0	1.0	1.0	1.0	0
4	7.0	35.0	14.0	80.0	2.0	15.0	0.0	1.0	0.0	0

Test

Actual Labels y

Predicted churn	LogLoss
0.91	0.11
0.13	2.04
0.04	0.04
0.23	0.26
0.43	0.56

$LogLoss = 0.60$

$\hat{y}$ Predicted Probability

$$LogLoss = -\frac{1}{n}\sum(y \times \log(\hat{y}) + (1-y) \times \log(1-\hat{y}))$$

LogLoss: $\boxed{0.00}$ ... 0.35 .... 0.60 ... $\boxed{1.00}$

Higher Accuracy