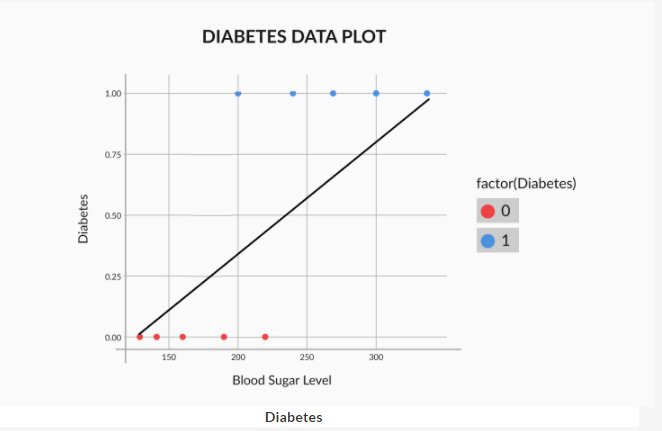


since the **sigmoid curve**has all the properties you would want — extremely low values in the start, extremely high values in the end, and intermediate values in the middle — it’s a good choice for modelling the value of the **probability of diabetes**.

This is the sigmoid curve equation: y=P(Diabetes)=1/1+e−(β0+β1x). Here, let’s say you take β0 = -15 and β1 = 0.065. Now, what will be the probability of diabetes for a patient with sugar level 220?

**You answered : 0.33 Correct Answer : 0.33**

**Feedback :**Here, the probability of diabetes for a person with sugar level x is given by $|#P(Diabetes)=\frac{1}{1+e^{-(β\_0+β\_1x)}}#|$. Now, taking $|#β\_0#|$ = -15 and $|#β\_1#|$ = 0.065, the probability of diabetes for a person with sugar level 220 will be given by $|#P = \frac{1}{1+e^{15-0.065\*220}} = 0.33#|$.



The main problem with a straight line is that it is not steep enough. In the sigmoid curve, as you can see, you have low values for a lot of points, then the values rise all of a sudden, after which you have a lot of high values. In a straight line though, the values rise from low to high very uniformly, and hence, the “boundary” region, the one where the probabilities transition from high to low is not present.

So, the best fitting combination of β0β0 and β1β1 will be the one which maximises the product:

(1−P1)(1−P2)(1−P3)(1−P4)(1−P6)(P5)(P7)(P8)(P9)(P10)

This product is called the **likelihood function**. It is the product of:

[(1−Pi)(1−Pi)(1−Pi)(1−Pi)------ for all non-diabetics --------] \* [(Pi)(Pi)(Pi)(Pi) -------- for all diabetics -------]

So, say that for the ten points in our example, the labels are a little different, somewhat like this:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Point no. | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Diabetes | no | no | no | yes | no | yes | no | yes | yes | yes |

In this case, the likelihood would be equal to (1−P1)(1−P2)(1−P3)(1−P5)(1−P7)(P4)(P6)(P8)(P9)(P10)

If you had to find β0andβ1 for the best fitting sigmoid curve, you would have to try a lot of combinations, unless you arrive at the one which maximizes the likelihood. This is similar to linear regression, where you vary β0 and β1 until you find the combination that minimizes the cost function. Hence, this is called a Generalized Linear regression Model (**GLM**), or a **logistic regression** model.

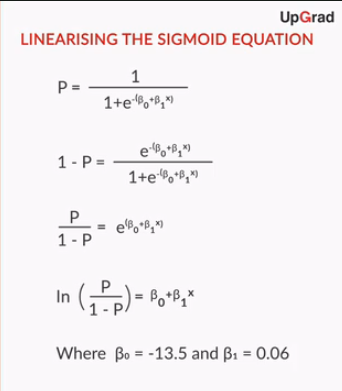
 that the best fit sigmoid curve would be the one that corresponds to the maximum likelihood value. In other words, the combination of betas that corresponds to the maximum value of likelihood will be the best fit combination.

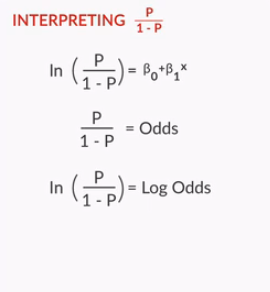
So far, you’ve seen this equation for logistic regression:

P=1/1+e−(β0+β1x)

Recall that this equation gives the relationship between P, the probability of diabetes and x, the patient’s blood sugar level.

While the equation is correct, it is not very intuitive. In other words, the relationship between P and x is so complex that it is difficult to understand what kind of trend exists between the two. If you increase x by regular intervals of, say, 11.5, how will that affect the probability? Will it also increase by some regular interval? If not, what will happen?





**Odds**

So, from the equation P=1/1+e−(β0+β1x) which tells you the probability of a person having diabetes, you got the following equation:

ln(P1−P)=β0+β1x

where the term (P/1-P) is referred to as the odds of a person having diabetes. What do the odds signify?

Ans#

**The odds tell you how much more likely the person is to have diabetes than he/she is to not have diabetes**

**Feedback :**P is the probability of a person having diabetes. So naturally, (1-P) is the probability of that person not having diabetes. Now, say (P/1-P) = 4. This would basically mean that the person is 4 times more likely to be a diabetic than he/she is to not be a diabetic.

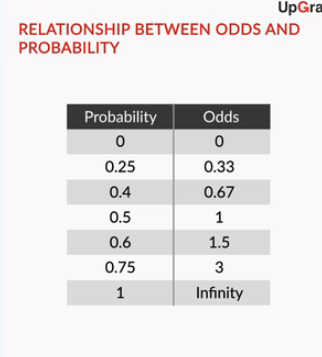
Let’s say that for a specific person, P(Diabetes) = 0.5. What are the odds of that person having diabetes?

Odds are given by P/1-P. Here, P = 0.5. Now, by putting in the value of P, you get that the odds, i.e. P/1-P, are equal to 0.5/0.5 = 1. Both classes are equally likely

Let’s say that for a specific person, Odds(Diabetes) = 2. What is the probability of that person having diabetes?

Recall that the odds are given by P/1-P. So, here, P/1-P = 2, which gives P = 2(1-P). Simplifying this further, you get 3P = 2, i.e. P = 2/3.

# the relationship between x and log odds is linear



So, the relationship between x and probability is not intuitive, while that between x and **odds/log odds** is. This has important implications. Suppose you are discussing sugar levels and the probability they correspond to. While talking about 4 patients with sugar levels of 180, 200, 220 and 240, you will not be able to intuitively understand the relation between their probabilities (10%, 28%, 58%, 83%). However, if you are talking about the log odds of these 4 patients, you know that their log odds are in a **linearly increasing pattern** (-2.18, -0.92, 0.34, 1.60) and that the odds are in a **multiplicatively increasing pattern** (0.11, 0.40, 1.40, 4.95, increasing by a factor of 3.55).

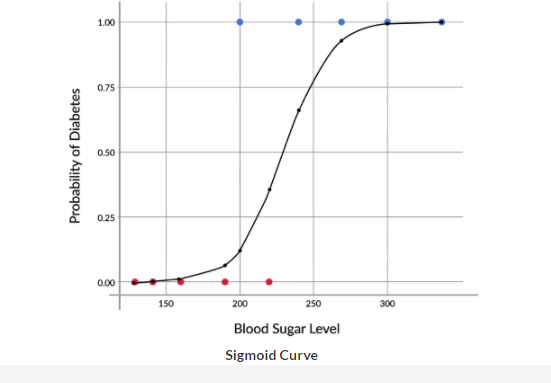
**Summary**

**binary classification** is. Basically, it is a classification problem in which the target variable has only 2 possible values.

**diabetes example** , wherein you tried to predict whether a person has diabetes or not based on that person’s blood sugar level.

You saw why a **simple boundary decision approach** does not work very well for this example. It would be too risky to decide the class blatantly on the basis of the cutoff because, especially in the middle, the patients could belong to any class — diabetic or non-diabetic.

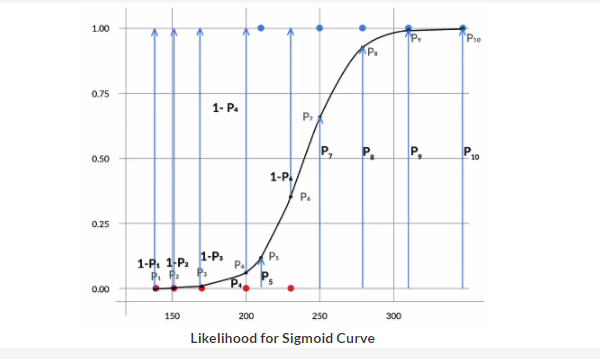
Hence, you learnt that it is better to talk in terms of **probability**. One such curve which can model the probability of diabetes very well is the **sigmoid curve**.



Its equation is given by the following expression:

P(Diabetes)=1/1+e−(β0+β1x)

Then, you learnt that in order to find the **best-fit sigmoid curve**, you need to vary β0 and β1 until you get the combination of beta values that maximises the **likelihood**. For the diabetes example, the likelihood is given by the expression:



Likelihood=(1−P1)(1−P2)(1−P3)(1−P4)(P5)(1−P6)(P7)(P8)(P9)(P10)

It is the product of:

[(1−Pi)(1−Pi)(1−Pi)(1−Pi) ------ for all non-diabetics --------] \* [(Pi)(Pi)(Pi)(Pi) -------- for all diabetics -------]

This process, where you vary the betas until you find the best fit curve for the probability of diabetes, is called **logistic regression.**

This process, where you vary the betas until you find the best fit curve for the probability of diabetes, is called **logistic regression.**

After this, you saw a simpler way of interpreting the equation for logistic regression. You saw that the following linearised equation is much easier to interpret:

ln(P1−P)=β0+β1x

The left-hand side of this equation is what is called **log odds**. Basically, the **odds** of having diabetes (P/1-P), indicate how much likelier a person is to have diabetes than to not have it. For example, a person for whom the odds of having diabetes are equal to 3, is 3 times more likely to have diabetes than to not have it. In other words, P(Diabetes) = 3 \* P(No diabetes).

You also saw how odds vary with variation in x. Basically, with every **linear increase** in x, the increase in odds is **multiplicative**. For example, in the diabetes case, after every increase of 11.5 in the value of x, the odds are approximately doubled, i.e. they increase by a multiplicative factor of about 2.

Dataset (mtcars)

So, suppose you build a model in R using glm (), that predicts the variable "am" based on variable "cyl" . The coefficients for the model thus obtained will be:

**New Bitmap Image.png**

Now, for a car with 4 cylinders, the probability of it having a manual transmission is closest to:

According to the model obtained earlier, the probability of a car having manual transmission is given by P=11+e−(3.777−0.691x)P=11+e−(3.777−0.691x) , where x is the number of cylinders in the car. So, for a car that has 4 cylinders, the required probability is equal to 11+e−(3.777−0.691∗4)11+e−(3.777−0.691∗4)= 0.73, or 73%.

For a car with 4 cylinders, the log odds of it having a manual transmission are closest to:

Recall that, for a logistic regression model, log odds are simply equal to the linear term, i.e. log odds = β0+β1xβ0+β1x. Here, β0β0= 3.777 and β1β1= -0.691. Also, x, the number of cylinders is equal to 4. Putting in all these values, you will get that the log odds of manual transmission are equal to 3.777 + (-0.691\*4) = 1.013. Clearly, it is much easier to calculate the log odds than it is to calculate the probability

For a car with 4 cylinders, the odds of manual transmission are equal to 2.76. For a car with 5 cylinders, the odds are equal to 1.38. For a car with 6 cylinders, the odds are closest to:

Recall that the odds increase multiplicatively with an increase in x. In this case, after x increased by 1, from 4 to 5, the odds halved from 2.76 to 1.38, i.e. multiplied by ½. Now, if x increases by 1 more, i.e. from 5 to 6, the odds will get halved again, from 1.38 to 0.69.

Telecom use case

Recall that, for **continuous variables**, the scale command is used to **standardise** the scales for three continuous variables — tenure, monthly charges and total charges. What the scale command basically does is — it converts values to the z-scores.

For example, let’s say that, for a particular customer, tenure = 72. After standardising, the value of scaled tenure becomes 72−32.424.6=1.6172−32.424.6=1.61, because for the variable tenure, mean(μ)= 32.4 and standard deviation(σ)= 24.6.

The variables had these ranges before standardisation:

1. Tenure = 1 to 72
2. Monthly charges = 18.25 to 118.80
3. Total charges = 18.8 to 8685

After standardisation, the ranges of the variables changed to:

1. Tenure = -1.28 to +1.61
2. Monthly charges = -1.55 to +1.79
3. Total charges = -0.99 to 2.83

The useful variables are selected from the model and discards the rest using a **stepwise AIC** algorithm.

insignificant variables and multicollinear ones are removed from the model on the basis of **VIF and p-value**.

Now, suppose you are a data analyst working for the company, and your boss has asked you to compare two of the customers, customer A and customer B. For both of them, the value of the variables tenure, PhoneService, Contract.xOne.year, etc. are all the same, except for the variable PaperlessBilling, which is equal to 1 for customer A and 0 for customer B.

In other words, customer A and customer B have the exact same behaviour as far as these variables are concerned, except that customer A opts for paperless billing, and customer B does not.

Based on the above information, what can you say about the log odds of these two customers?



**log odds (customer A) > log odds (customer B)**

**Feedback :**Recall that log odds are basically equal to the linear term in logistic regression. So, here, you logodds=β0+β1x1+β2x2+β3x3+......+β10x10. Now, for the two customers, all beta and all x values are the same, except for x5x5, which is equal to 1 for customer A and 0 for customer B. So, the log odds for customer A will exceed those of customer B by 5, which is equal to 0.34.

Now what can you say about the odds of churn for these two customers?



**For customer A, the odds of churning are higher than for customer B**

**Feedback :***Recall that in the last question, you were told that log odds for customer A are higher than those for customer B. So, the odds of churning for customer A are also higher than the odds of churning for customer B. This is because, as the number increases, its log increases and vice versa.*

# Summary

In this session, you learnt how to **build a multivariate logistic regression model in R**. The equation for multivariate logistic regression is basically just an extension of the univariate equation:

P=11+e−(β0+β1x1+β2x2+β3x3+....)P=11+e−(β0+β1x1+β2x2+β3x3+....)

The example used for building the multivariate model in R was the **telecom churn example**. Basically, you learnt how R can be used to decide the probability of a customer churning based on the value of 21 predictor variables such as monthly charges, paperless billing, etc.

First, the data was imported, which was present in 3 separate csv files. After creating a merged master data set, one that contains all 21 variables, **data preparation** was done, which involved the following steps:

1. Missing value imputation
2. Outlier treatment
3. Standardising scales of continuous variables
4. Dummy variable creation for categorical variables

After all of this was done, a logistic regression model was built in R using the function **glm()**. This model contained all the variables, some of which had insignificant coefficients; for many of them, the coefficients were NA. Hence, some of these variables were removed using these two algorithms:

1. **Stepwise variable selection** based on AIC [using stepAIC()]
2. **Backward variable selection** based on VIF and p-value

After this whole process was completed, you ended up with **10 variables**, all of which were significant. Also, the VIF for any variable was not very high. Thus, this model was declared as the final model.

Since you have seen most of the concepts in this lesson earlier, such as dummy variables, stepAIC(), VIF and p-value, the session went at a faster pace from here. However, if you need to, you can always look at the R code provided to you in more detail.

Also, EDA, missing value imputation, and outlier treatment, which are generally the first steps that is executed on getting the data, were not covered at all. If you still want to take a closer look at it, you can again find it in your R code.

Model Evaluation

However, every model has to be checked in order to understand how well it is doing. This process, as you may recall, is called model evaluation. In a **model evaluation** process, you take your model that has been prepared with the help of training data, and you use it to make **predictions for the testing data**. After that, since you have the actual values for the testing data too, you can compare the actual values with the predicted values, and that will help you understand how well your model is doing.

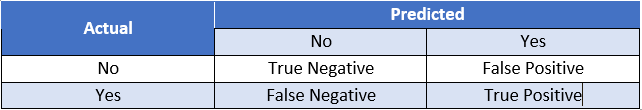
So, now you know what sensitivity and specificity are. To recap:

* Sensitivity(ProportionofYesescorrectlypredicted)=NumberofactualYesescorrectlypredicted/TotalnumberofactualYeses
* Specificity(ProportionofNoscorrectlypredicted)= NumberofactualNoscorrectlypredicted/TotalnumberofactualNos

For the telecom churn example, since yes corresponds to churn and no corresponds to non-churn, you can write:

* Sensitivity=Numberofactualchurnscorrectlypredicted/Totalnumberofactualchurns
* Specificity=Numberofactualnon−churnscorrectlypredicted/Totalnumberofactualnon−churns

To summarise:



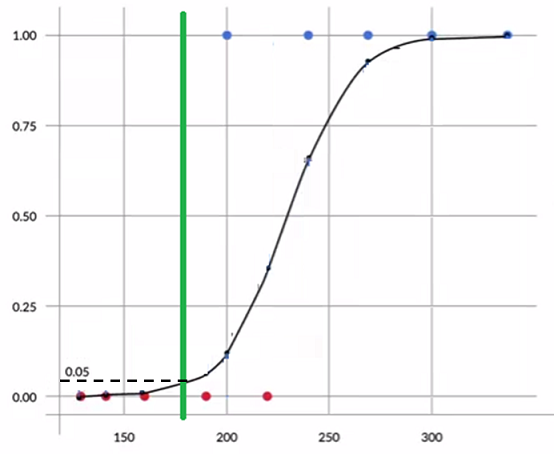
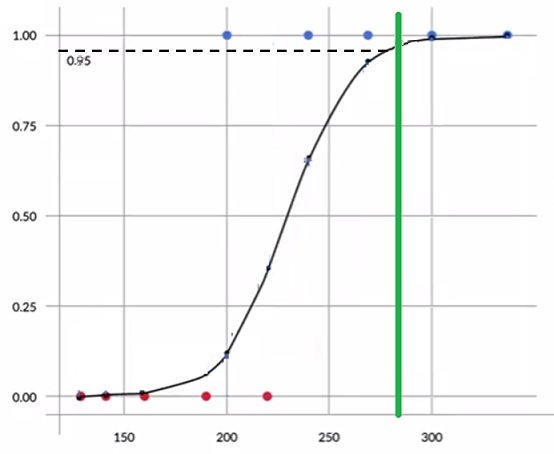
**General Confusion Matrix**

1. True Negatives (TN) are actual negatives, correctly predicted as negatives
2. False Negatives (FN) are actual positives, incorrectly predicted as negatives
3. True Positives (TP) are actual positives, correctly predicted as positives
4. False Positives (FP) are actual negatives, incorrectly predicted as positives

Now, you can revisit the definition of sensitivity and specificity.

* Sensitivity(TruePositiveRate)=TPTP+FNSensitivity(TruePositiveRate)=TPTP+FN
* Specificity(TrueNegativeRate)=TNTN+FPSpecificity(TrueNegativeRate)=TNTN+FP

Also, you can talk about what is called the FPR (False Positive Rate) and FNR (False Negative Rate).

* FPR(FalsePositiveRate)=FPFP+TNFPR(FalsePositiveRate)=FPFP+TN
* FNR(FalseNegativeRate)=FNFN+TP
* Lastly, you can also write this expression for accuracy:
* Accuracy=TP+TNTP+FP+TN+FN
* Now, as you may have noticed, when the cutoff of a model is increased, its sensitivity decreases and its specificity increases. Why does that happen? In order to understand that, you’ll have to go all the way back to the diabetes example.
* Recall the best fit curve for the diabetes example. Now, for that curve, suppose you take a low cutoff, like 0.05. Graphically, it would look like this:
* 
* **Diabetes Sigmoid Curve (Cutoff = 0.05)**
* So here, all points after the green line would be declared diabetics. Conversely, all points before the green line would be declared non-diabetics.
* What this low cutoff basically means is that, even if a patient has a minuscule chance of being diabetic, that patient is predicted to be a diabetic. Because of this approach, all the diabetic patients have been correctly identified as diabetic, i.e. the sensitivity of the model is pretty high. However, a patient is predicted as non-diabetic, only if the corresponding probability of diabetes is very very low. Because of that, the non-diabetic patients have not been identified with the same accuracy, i.e. the **specificity of the model is not high**.
* Now, what will happen if the cutoff is very high, say, 0.95? Graphically, it looks like this:
* 
* **Diabetes Sigmoid Curve (Cutoff = 0.95)**
* Here you have a high cutoff. What this basically means is that, unless a patient has a very very high chance of being diabetic, that patient is predicted to be a non-diabetic. Because of this approach, all the non-diabetic patients have been correctly identified as non-diabetic, i.e. the specificity of the model is pretty high. However, a patient is predicted as diabetic only if the corresponding probability of diabetes is very very high. Because of that, the diabetic patients have not been identified with the same accuracy, i.e. the **sensitivity of the model is not high**.

Based on your understanding till now, what are the two main differences between logistic regression and linear regression?

*The two main important differences between logistic and linear regression are: 1. Dependent/response variable in linear regression is continuous whereas, in logistic regression, it is the discrete type. 2. Cost function in linear regression minimise the error term (y-y2) but logistic regression uses maximum likelihood method for maximising probabilities.*

logistic regression is a widely used technique in various types of industries. This is because of two **main** **reasons**:

1. It is very easy to **understand** and offers an **intuitive explanation** of the variables
2. The output (i.e. the probabilities) has a linear relationship with the log of odds, which can be very useful for explaining results to managers
3. let’s go back to the telecom churn example from earlier sessions:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **customerID** | **Probability** | **Odds** | **Log Odds** | **Score** |
| 8773-HHUOZ | 0.084 | 0.092 | -2.389 | 331 |
| 8865-TNMNX | 0.257 | 0.346 | -1.062 | 369 |
| 9867-JCZSP | 0.297 | 0.422 | -0.862 | 375 |
| 9420-LOJKX | 0.435 | 0.770 | -0.261 | 392 |
| 6234-RAAPL | 0.439 | 0.783 | -0.245 | 393 |
| 7760-OYPDY | 0.443 | 0.795 | -0.229 | 393 |
| 8012-SOUDQ | 0.446 | 0.805 | -0.217 | 394 |
| 3413-BMNZE | 0.461 | 0.855 | -0.156 | 395 |
| 6575-SUVOI | 0.688 | 2.205 | 0.791 | 423 |
| 6388-TABGU | 0.753 | 3.049 | 1.115 | 432 |

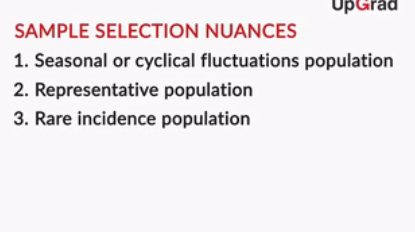
You must have noticed the column called score. Basically, it’s a different way of reporting your findings. Earlier, you saw that log odds make more sense as the output instead of probabilities because of their linear relationship with the variables. However, log odds have weird values, such as -0.245, -0.156 etc., which is not a very elegant form of output.

Hence, instead of reporting the log odds as output, you can report scores.

Score is calculated using the following expression:

Score=400+(20∗log(odds)log(2))Score=400+(20∗log(odds)log(2))

This **expression** is decided based on business understanding. You could come up with your own expression for the score, one that converts log odds into a more presentable form



1. **Cyclical** or**seasonal fluctuations** in the business that need to be taken care of while building the samples. E.g. Diwali sales, economic ups and downs, etc.
2. The sample should be **representative of the population** on which the model will be applied in the future.
3. For **rare events samples**, the sample should be balanced before it is used for modelling.

 Let's talk about the ICICI credit card example again.

For students and salaried people, different variables may be important. While students' defaulting and not defaulting will depend on factors such as program enrolled for, the prestige of the university attended, parents' income, etc., the probability of salaried people will depend on factors such as marital status, income, etc. So, the predictive pattern across these two segments is very different, and hence, it would make more sense to make different child models for both of them, than to make one parent model.

categorical variables have to be transformed into dummies. Also, you were told that numeric variables have to be standardised, so that they all have the same scale. However, you could also convert numeric variables into dummy variables

There are some pros and cons of transforming variables to dummies. Creating dummies for **categorical variables** is very straightforward. You can directly create n-1 new variables from an existing categorical variable if it has n levels. But for **continuous variables**, you would be required to do some kind of EDA analysis for binning the variables

There are some pros and cons of transforming variables to dummies. Creating dummies for **categorical variables** is very straightforward. You can directly create n-1 new variables from an existing categorical variable if it has n levels. But for **continuous variables**, you would be required to do some kind of EDA analysis for binning the variables.

The **major advantage** offered by **dummies** especially for continuous variables is that they make the **model stable**. In other words, small variations in the variables would not have a very big impact on a model that was made using dummies, but they would still have a sizeable impact on a model built using continuous variables as is.

Let’s now move on to another technique commonly used for transforming variables — **Weight of evidence (WOE) analysis**.

So, to summarise, you learnt **three important** things in this lecture:

1. Calculating **woe** **values** for fine binning and coarse binning
2. The **importance** of woe for fine binning and coarse binning
3. The **usage** of woe transformation

**WOE** can be calculated using the following equation:

 WOE = ln(good in the bucketTotal  Good )−ln(bad in the bucketTotal  bad )  WOE = ln⁡(good in the bucketTotal  Good )−ln⁡(bad in the bucketTotal  bad )

Or, it can be expressed as:

WOE=ln(PercentageofGoodPercentageofBad)WOE=ln(PercentageofGoodPercentageofBad)

Once you've calculated woe values, it is also important to note that they should follow an **increasing or decreasing trend** across bins. If the trend is not **monotonic**, then you would need to compress the buckets/ bins (coarse buckets) of that variable and then calculate the WOE values again.

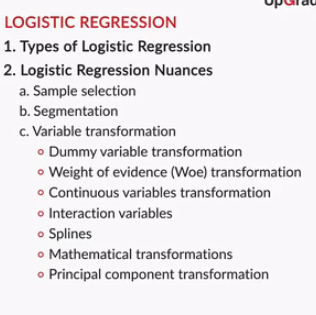
As mentioned in the lecture, there are two main advantages of WOE:

1. WOE reflects group identity: This means it captures the general trend of distribution of good and bad customers. E.g. the difference between customers with 30% credit card utilisation and 45% credit card utilisation is not the same as the difference between customers with 45% credit card utilisation and customers with 60% credit card utilisation. This is captured by transforming the variable credit card utilisation using WOE.
2. WOE helps you in treating missing values logically for both types of variables — categorical and continuous. E.g. in the credit card case, if you replace the continuous variable credit card utilisation with WOE values, you would replace all categories mentioned above (0%-45%, 45% - 60%, etc.) with certain specific values, and that would include the category "missing" as well, which would also be replaced with a WOE value.

So, basically, the pros and cons of a WOE transformation are similar to dummy variables.

1. Pros: The model becomes more stable because small changes in the continuous variables will not impact the input so much.
2. Cons: You may end up doing some score clumping.

This is because when you are using WOE values in your model, you are doing something similar to creating dummy variables — you are replacing a range of values with an indicative variable. It is just that, instead of replacing it with a simple 1 or 0, which was not thought out at all, you are replacing it with a well thought out WOE value. Hence, the chances of undesired score clumping will be a lot less here.



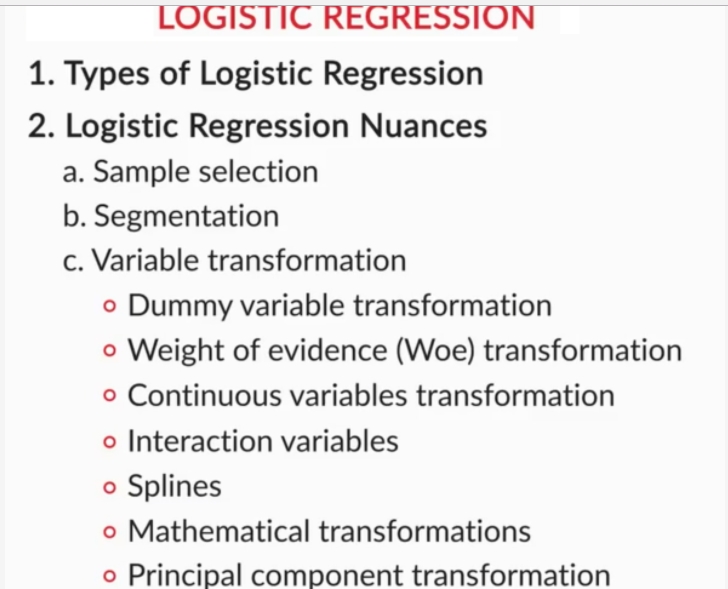
Another kind of transformation that is usually done is a spline. A spline is basically obtained by fitting a polynomial over the WOE values. Using a spline offers high predictive power, but it may result in unstable models.

Mathematical transformations such as x2x2, x3x3 and logxlogx are also commonly used. However, these result in variables that are not easy to explain. For example, the fact that (income)2(income)2, not incomeincome, is a good predictor of default, is not very intuitive and, hence, difficult to explain.

Principal Component Transformation is also a very important and very effective transformation. It transforms variables into components that are orthogonal to each other. Another way of saying that is — the variables are

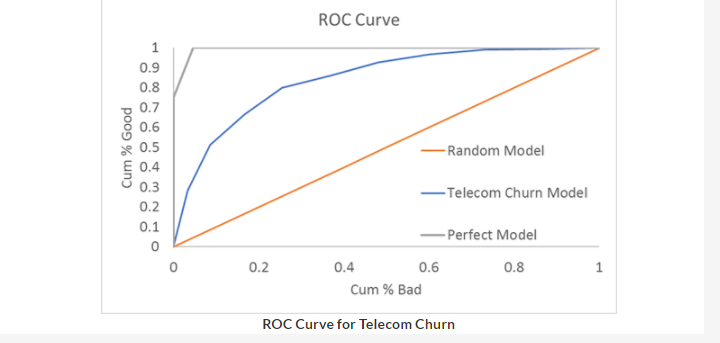
grouped and modified and put into a few "bunches" such that each "bunch" is not correlated with the other.

So, PCA (Principal Component Analysis) is very powerful since it already ensures that multicollinearity in the model is very low. However, the problem with PCA, just like with other advanced transformations, is that the components do not offer a very intuitive or easily explainable interpretation — the model's inner workings would be hard to communicate to a business manager.



Let's look at 2 cases, where the variable's value is missing (equal to NA):

1. **Utilisation is missing:**As mentioned earlier, if this variable is missing for a particular customer, that could very well be because the bank did not find that customer worthy enough for a credit card. Hence, these missing values are not missing at random, and it would be unfair to just replace them with the mean or the median. As mentioned earlier, it would be wiser to perform a WOE analysis and then replace these values.
2. **Age is missing:**Consider why the variable age is missing for some customers. Here, it may actually make more sense to just replace the missing value with the mean or the median, instead of wasting time on WOE analysis. This is because it is very likely that the variable age is just missing because of a system error or a manual error, and there is no clear pattern behind the missing values.



The above image also contains the ROC curve for the random model (red line) and the perfect model (grey line).

Clearly, the perfect model is pretty much a right triangle, whereas the random model is a straight line. Basically, a model that rises steeply is a good model.

Another way of saying that is — it will have a higher area under the curve. So, the **gini** coefficient, which is given by:

Gini=AreaUnderROCCurveGini=AreaUnderROCCurve

will be high for a good model.

The data used to build the model was from 2014.

You split the original data into two parts, i.e. training and testing data. However, these two parts were both built with data from 2014.



**Training and Testing Split**

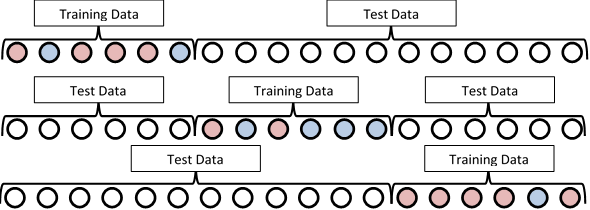
This is called in-sample validation. Testing your model on this test data may not be enough though, as the test data here is too similar to training data.

So, it makes sense to actually test the model on data that is from some other time, like 2016. This is called out-of-time validation.



**Out of Time Validation**

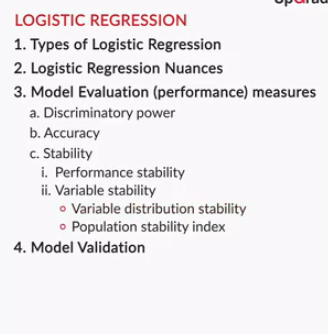
Another way to do the same thing is to use K-fold cross validation. Basically, the evaluation of the sample is done for k-iterations. E.g. here's a representation of how 3-fold cross validation works:



**3 Fold Cross Validation**

Basically, there are 3 iterations in which evaluation is done. In the first iteration, 1/3rd of the data is selected as training data and the remaining 2/3rd of it is selected as testing data. In the next iteration, a different 1/3rd of the data is selected as the training data set and then the model is built and evaluated. Similarly, the third iteration is completed.

Such an approach is necessary if the data you have for model building is very small, i.e. has very few data points.

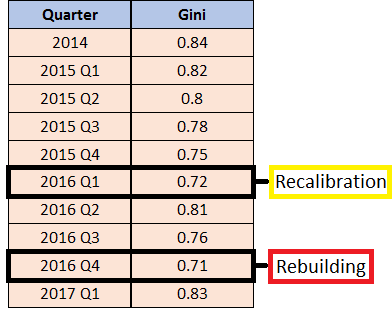


Obviously, a good model will be stable. A model is considered stable if it has:

1. **Performance Stability:** Results of in-sample validation approximately match those of out-of-time validation
2. **Variable Stability:** The sample used for model building hasn't changed too much and has the same general characteristics



Let's go back to the telecom churn example. If you recall, the model was built using **data from 2014**. Now suppose, you are tracking its performance over time, and that ends up giving you the following results:



**Model Performance Over Time**

So, the **first** **time**, when the model's Gini dropped to **0.72**, you avoided building a new model. Basically, you just **recalibrated**, i.e. updated the coefficients of the variables. That resulted in a slight increase of Gini. However, the next time Gini dropped to a low value, i.e. **0.71**, we just rebuilt the model, i.e. got new sample data, performed data prep, etc. and built the entire model.

In this session, you learnt about various model evaluation measures, such as accuracy, sensitivity, specificity, KS statistic, etc as shown in figure 1. Then, you learnt how to validate your model on in-sample data and out-time data (unseen data). Also, for stability, you learnt and understood stability by two metrics — Variable Distributive Stability and Predictive Pattern Stability. Then, you learnt about model tracking and model recalibrating steps in detail.

