Contents

[Logloss and weighted logloss 2](#_Toc521344023)

[Classification Table /Confusion Matrix. 2](#_Toc521344024)

[Sensitivity (TP) 3](#_Toc521344025)

[Specificity (TN) 3](#_Toc521344026)

[Accuracy 3](#_Toc521344027)

[Misclassification rate 3](#_Toc521344028)

[Precision and Recall 3](#_Toc521344029)

[Area Under ROC Curve. 4](#_Toc521344030)

[Gini Coefficient 5](#_Toc521344031)

[KS statistics 5](#_Toc521344032)

[Lift Curve 5](#_Toc521344033)

[What is Lift Curve 5](#_Toc521344034)

[How to interpret Lift Curve 7](#_Toc521344035)

[Which Lift curve is good 7](#_Toc521344036)

[Steps to build the Lift Curve 8](#_Toc521344037)

[Use of Lift Chart 11](#_Toc521344038)

[Interview Questions 11](#_Toc521344039)

[What error metric would you use to evaluate how good a binary classifier is? What if the classes are imbalanced? What if there are more than 2 groups? 11](#_Toc521344040)

[I have two models of comparable accuracy and computational performance. Which one should I choose for production and why? 11](#_Toc521344041)

[How can you prove that one improvement you've brought to an algorithm is really an improvement over not doing anything? 11](#_Toc521344042)

[Is it better to have too many false positives, or too many false negatives? Explain 12](#_Toc521344043)

Classification problems are perhaps the most common type of machine learning problem and as such there are a myriad of metrics that can be used to evaluate predictions for these problems.

Logloss and weighted logloss: This metric gives an intuition how well we can trust our predicted probabilities. Logloss is not good for imbalanced classes but

weighted logloss is, as more weight is given to minor class predictions.

# Classification Table /Confusion Matrix.

1. Error Rate = (FP+FN)/(P+N)
2. Accuracy = (TP+TN)/(P+N)
3. Sensitivity(Recall or True positive rate) = TP/P

Matters more when classifying 1 is more imp

1. Specificity(True negative rate) = TN/N

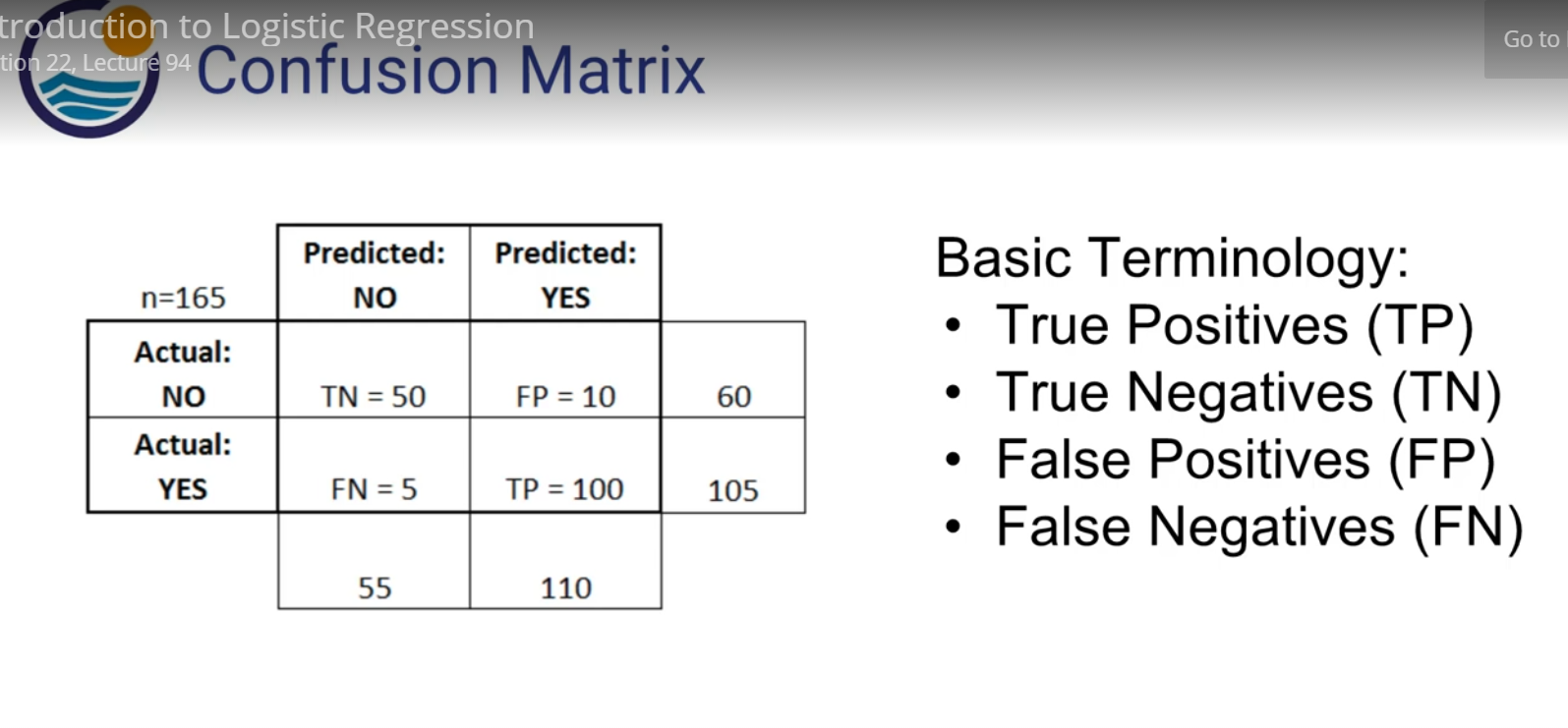
Matters more when classifying 0 is more imp

1. False Positive Rate = 1-Specificity (FP/N)
2. Precision(Positive predicted value) = TP/(TP+FP)

Precision = TP/Predicted Positive

1. F-Score(Harmonic mean of precision and recall) = (1+b)(PREC.REC)/(b^2PREC+REC) where b is commonly 0.5, 1, 2.

* **We use a confusion matrix to evaluate our model**



**FP 01 (True - Predicted) – predicted yes ,but in reality they don’t have disease - Type 1 error**

So False positives are the false predicted positives but inTrue they are 0

**FN (10) - predicted no , but in reality that is yes– Type 2 erro r**

**TP – (11)**

**TN – (00)**

**Accuracy = (tp + tn)/ total**

**Error rate = (fp + fn)/total**

## Sensitivity (TP)/recall/sensitivity

Sensitivity is the **true positive rate** also called the **recall**.

It is the number instances from the positive (first) class that are predicted correctly

Sensitivity = **TP/Actual P** = TP / TP +FN

11 /( 11 + 10)

Sensitivity matters more when classifying the 1’s correctly is more important than classifying the 0’s.

## Specificity (TN)

* Specificity is also called the **true negative rate**. Is the number of instances from the negative class (second) class that were actually predicted correctly.

Specificity = TN (Predicted 0) /N(Actual 0 ) = TN ( 00) / TN (00) +FP ( 01)

## False positive Rate

1. Specificity

Specificity matters more when classifying the 0’s correctly is more important than classifying the 1’s.

Maximizing specificity is more relevant in cases like spam detection, where you strictly don’t want genuine messages (0’s) to end up in spam (1’s).

## Accuracy

Number of correct prediction (TRUE POS + TRUE NEG) divided by sample size.

Misclassification rate

## Precision and Recall

The **precision** is the ratio tp / (tp(11) + fp (01) ) where tp is the number of true positives and fp the number of false positives. The precision is intuitively the ability of the classifier to not label a sample as positive if it is negative.

A high precision score gives more confidence to the model’s capability to classify 1’s.

True positive/ Predicted positive

The **recall** is the ratio tp / (tp + fn(10) where tp is the number of true positives and fn the number of false negatives. The recall is intuitively the ability of the classifier to find all the positive samples.

Recall = TP/ Actual P

Recall gives an idea of how many of the total 1’s it was able to cover.

A good model should have a good precision as well as a high recall. So ideally, I want to have a measure that combines both these aspects in one single metric – **the F1 Score**.

**F1 Score = (2 \* Precision \* Recall) / (Precision + Recall)**

The **F-beta** score can be interpreted as a weighted harmonic mean of the precision and recall, where an F-beta score reaches its best value at 1 and worst score at 0. The F-beta score weights the recall more than the precision by a factor of beta. beta = 1.0 means recall and precision are equally important.

# Area Under ROC Curve.

Sensitivity(Recall or True positive rate) = TP/P

Matters more when classifying 1 is more imp

Specificity(True negative rate) = TN/N

Matters more when classifying 0 is more imp

False Positive Rate = 1-Specificity (FP/N)

[Area Under the Receiver Operating Characteristic curve](https://en.wikipedia.org/wiki/Receiver_operating_characteristic#Area_under_the_curve).

**For a binary classifier, the ROC curve plots the true positive rate versus the false positive rate**, over a varying threshold. For example, say you developed some test for a disease. You would obtain some data on patients with the disease, and control subjects, then fit a model on it; the purpose of the model is to predict disease status from some set of variables. The true positive rate is subjects who have the disease who are correctly identified as having it, and the false positive rate is subjects who don’t have the disease, but are identified as having it (by your model).

Theres some threshold for designating that a person has the disease. Say at or above a test value of .5, you consider that enough evidence to denote that person as having the disease. As you vary this threshold, the true and false positive rates change. This is the ROC curve.

The most concise interpretation in my opinion, is the AUC, the area under the (ROC) curve. It ranges from 0.5 (classification at random) to 1.0 (perfect classification). You can generally think of the AUC as the probability that your model will correctly classify a given subject into one of the two categories.

It ranges from 0.5 to 1, where 0.5 corresponds to the model randomly predicting the response, and a 1 corresponds to the model perfectly discriminating the response.

# [How to determine the optimal threshold for a classifier and generate ROC curve?](https://stats.stackexchange.com/questions/123124/how-to-determine-the-optimal-threshold-for-a-classifier-and-generate-roc-curve)

The choice of a threshold depends on the importance of TPR and FPR classification problem. For example, if your classifier will decide which criminal suspects will receive a death sentence, false positives are very bad (innocents will be killed!). Thus you would choose a threshold that yields a low FPR while keeping a reasonable TPR (so you actually catch some true criminals). If there is no external concern about low TPR or high FPR, one option is to weight them equally by choosing the threshold that maximizes TPR−FPR

# Relation between Sensitivity, Specificity, FPR and Threshold.

Sensitivity(True positive rate) and Specificity (True Negative Rate) are inversely proportional to each other. So when we increase Sensitivity, Specificity decreases and vice versa.

Sensitivity⬆️, Specificity⬇️ and Sensitivity⬇️, Specificity⬆️

When we decrease the threshold, we get more positive values thus it increases the sensitivity(True positive rate) and decreasing the specificity.

Similarly, when we increase the threshold, we get more negative values thus we get higher specificity and lower sensitivity.

As we know FPR is 1 - specificity. So when we increase TPR, FPR also increases and vice versa.

TPR⬆️, FPR⬆️ and TPR⬇️, FPR⬇️

# Gini Coefficient

Gini Coefficient is an indicator of how well the model outperforms random predictions. It can be computed from the area under the ROC curve using the following formula:

Gini Coefficient = (2 \* AUROC) – 1

# KS statistics

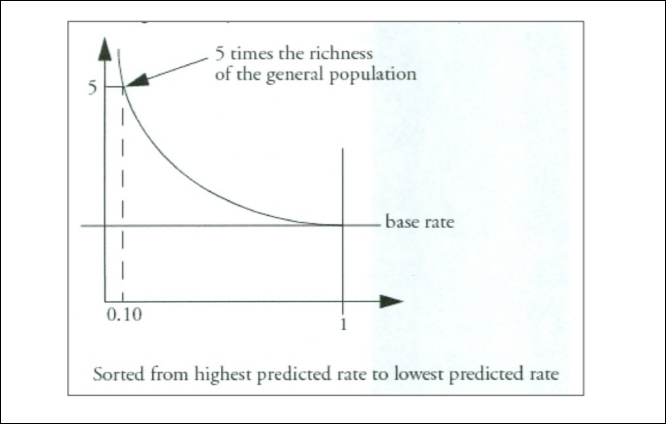
It looks at maximum difference between distribution of cumulative events and cumulative non-events

# Lift Curve

### What is Lift Curve

Lift Curves A lift curve shows the same information as an ROC curve, but in a way to dramatize the richness of the ordering at the beginning. The Y-axis shows the ratio of how rich that portion of the population is in the chosen response level compared to the rate of that response level as a whole.

For example, if the top-rated 10% of fitted probabilities have a 25% richness of the chosen response compared with 5% richness over the whole population, the lift curve would go through the X-coordinate of 0.10 at a Y-coordinate of 25% / 5%, or 5. All lift curves reach (1, 1) at the right, as the population as a whole has the general response rate.



Lets suppose there ae 60,000 customers total.

So 10 decile , in each decile there would be 6000 customers

There are 1000 responders total

So randomly (without model) = 1000/60,00 = 1.6%

So 96 in each of the decile

Now after the model we applied there are 400 responders in decile 1 (which is the top most decile)

Lift = Number of responders in each decile / Avg number of responders in each decile

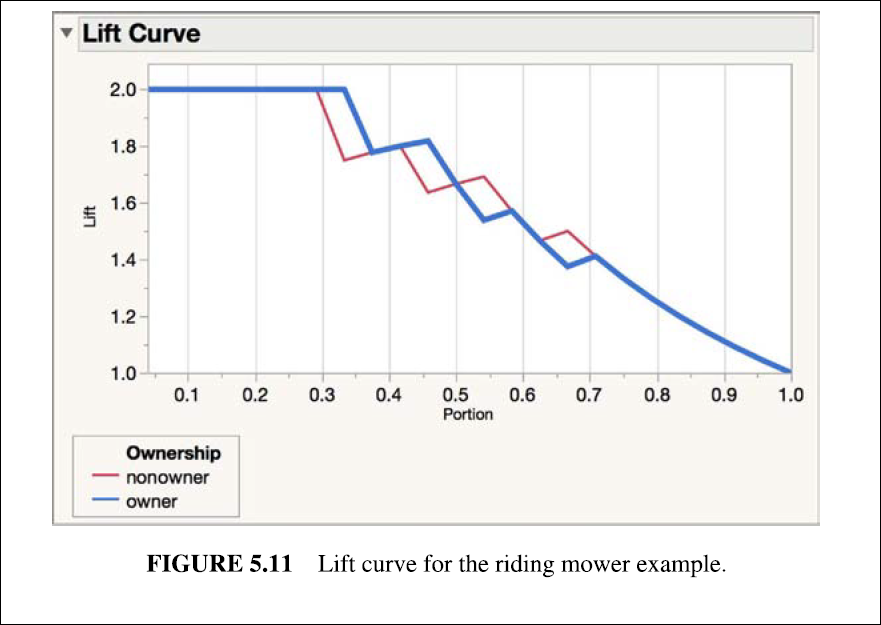
400/96\*100 =416% is the lift.

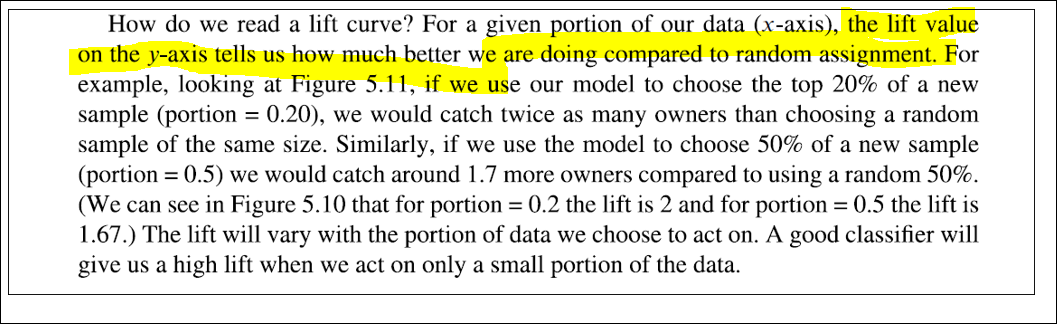
#### What Lift represents

So this number shows that there are 3.48% more responders than average.

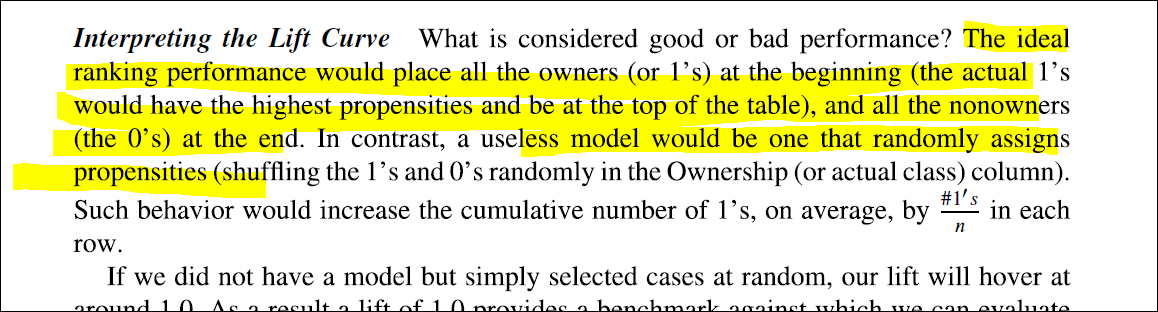
It also says that those in decile 1 who did not respond are very good targets .

### How to interpret Lift Curve





### Which Lift curve is good



### Steps to build the Lift Curve

### 

Gain and Lift chart are mainly concerned to check the rank ordering of the probabilities. Here are the steps to build a Lift/Gain chart:

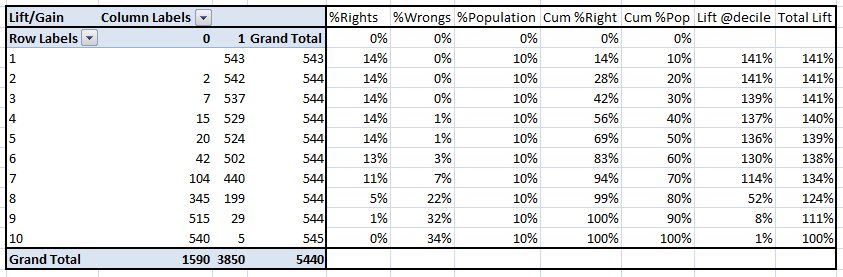
Step 1 : Calculate probability for each observation

Step 2 : Rank these probabilities in decreasing order.

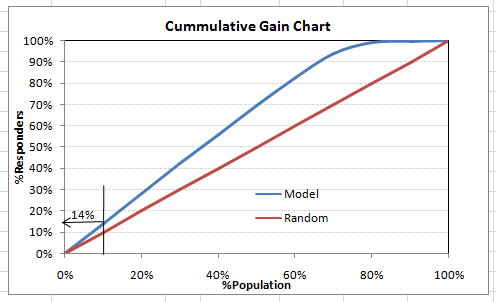
Step 3 : Build deciles with each group having almost 10% of the observations.

Step 4 : Calculate the response rate at each deciles for Good (Responders) ,Bad (Non-responders) and total.

You will get following table from which you need to plot Gain/Lift charts:

[](https://www.analyticsvidhya.com/blog/wp-content/uploads/2015/01/LiftnGain.png)

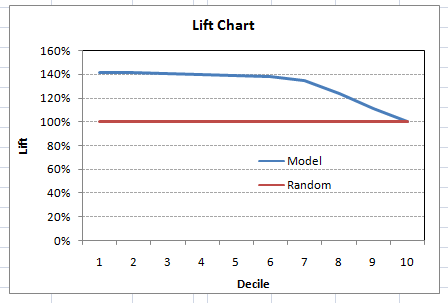
This is a very informative table. Cumulative Gain chart is the graph between Cumulative %Right and Cummulative %Population. For the case in hand here is the graph :

[](https://www.analyticsvidhya.com/blog/wp-content/uploads/2015/01/CumGain.png)

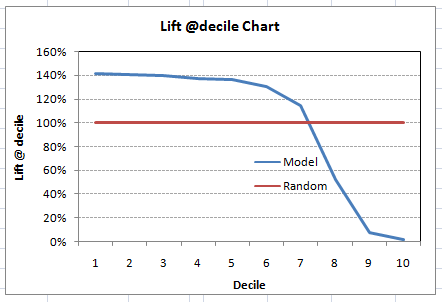
This graph tells you how well is your model segregating responders from non-responders. For example, the first decile however has 10% of the population, has 14% of responders. This means we have a 140% lift at first decile.

What is the maximum lift we could have reached in first decile? From the first table of this article, we know that the total number of responders are 3850. Also the first decile will contains 543 observations. Hence, the maximum lift at first decile could have been 543/3850 ~ 14.1%. Hence, we are quite close to perfection with this model.

Let’s now plot the lift curve. Lift curve is the plot between total lift and %population. Note that for a random model, this always stays flat at 100%. Here is the plot for the case in hand :

[](https://www.analyticsvidhya.com/blog/wp-content/uploads/2015/01/Lift.png)

You can also plot decile wise lift with decile number :

[](https://www.analyticsvidhya.com/blog/wp-content/uploads/2015/01/Liftdecile.png)

What does this graph tell you? It tells you that our model does well till the 7th decile. Post which every decile will be skewed towards non-responders. Any model with lift @ decile above 100% till minimum 3rd decile and maximum 7th decile is a good model. Else you might consider over sampling first.

### Use of Lift Chart

Lift / Gain charts are widely used in campaign targeting problems. This tells us till which decile can we target customers for an specific campaign. Also, it tells you how much response do you expect from the new target base.

It shows that those in top deciles who did not respond are very good targets, since again they all look alike. This is called clone modeling.

# Interview Questions

### What error metric would you use to evaluate how good a binary classifier is? What if the classes are imbalanced? What if there are more than 2 groups?

* Accuracy: proportion of instances you predict correctly. Pros: intuitive, easy to explain, Cons: works poorly when the class labels are imbalanced and the signal from the data is weak
* AUROC: plot fpr on the x axis and tpr on the y axis for different threshold. Given a random positive instance and a random negative instance, the AUC is the probability that you can identify who's who. Pros: Works well when testing the ability of distinguishing the two classes, Cons: can’t interpret predictions as probabilities (because AUC is determined by rankings), so can’t explain the uncertainty of the model
* logloss/deviance: Pros: error metric based on probabilities, Cons: very sensitive to false positives, negatives
* When there are more than 2 groups, we can have k binary classifications and add them up for logloss. Some metrics like AUC is only applicable in the binary case.

### I have two models of comparable accuracy and computational performance. Which one should I choose for production and why?

I would choose simpler ones

### How can you prove that one improvement you've brought to an algorithm is really an improvement over not doing anything?

You can always check the model performance after adding or removing a features, if the performance of model is dropping or improving you can see if the inclusion of that variable makes sense or not. Apart from that, you tweak different inbuilt model parameters like you increase number of trees to grow or number of iterations to do in random forest, you add a regularisation term in linear regression, you change threshold parameters in logistic regression, you assign weights to several algorithms , if you compare the accuracies and other statistics before and after making such change to model, you can understand if these result into any improvement or not.

### 

### Is it better to have too many false positives, or too many false negatives? Explain

Depends upon the problem we are solving. False positive is wrong prediction of something being absent, while it is actually present (1 classified as 0) while false negative is wrong prediction of something being present, while it is actually absent (0 classified as 1).   
Note:- 1 being something present, 0 being something absent.

Describe one example of a classification problem where the cost of a false positive (01) is way higher than false negative as well as the other way round

Whenever you tackle a data science project, one of the most important starting points is to have well clear whether the cost of a false positive is significantly higher, lower, or similar to the cost of a false negative. This will affect the model you choose, internal loss function, prediction cut-off point, and, possibly, even the training set itself.

A good example for large false positive cost is the recruiting process. In general, the cost of hiring a bad

candidate is much higher than passing on a good one.

Desired - Good

Actual - Bad

Predicted – Good

Problem statement is whether a person is good or bad

01 – False Positive Whether a person is bad but we thought he would be good

10 – False Negative – When the person would be good but we thought we was bad

A bad hiring will cost a lot of money in terms of not generating value, affecting the team morale, firing expenses, and the subsequent new recruiting process costs. Assuming you will have more than one good candidate in the pipeline within a reasonable time frame, passing on a good one is not that bad.

At this point, it would be interesting to draw a comparison between your current recruiting process and a machine learning classifier. You are interviewing with several people. Each interviewer can be seen as a classifier and the final classification will be given by a combination of each interviewer vote. This is exactly like ensemble methods. Furthermore, each interviewer will focus on a subset of your skills. Again, this is, for instance, similar to what happens in a Random Forest (RF). In a RF, each tree is built on bootstrap replicas of the original dataset. A consequence of this is that some events will get higher weight, so each tree can focus on learning more specific areas.

This is also a nice example since it will highlight how you consider the company where you are interviewing classical example of a large cost of false negatives is cancer detection. If you predict someone has cancer and it is not true, nothing major really happens. That person will have a bad few days until she finds out she is

healthy. The opposite can have dramatic consequences, since that person will not get the appropriate treatment.

Finally, please note that minimizing the cost of false positives/negatives is a machine learning problem as much as a product problem. It is extremely important to re-design the product experience in a way that reduces those costs. A common and effective strategy is to design three paths: one for people with low probability of being

case 1, one for people with high probability, and one for people in between.

For instance, in the recruiting case: the really good ones get an offer, the average and bad ones get rejected, and the good ones start with an internship. In risk, people predicted as "no fraud", go through the normal site experience. People predicted as "very high risk" get blocked. And people predicted as moderately high risk, go through an additional verification step live on the site.

## Explain what precision and recall are. How do they relate to the ROC curve?

Recall is sensitivity

Precision = TP/Predicted Positive