Classification: Performance Measures

M. R. Hasan
Readings

• Geron: Chapter 3
K-Nearest Neighbors Model

- So far we have discussed how to perform classification using the K-Nearest Neighbors model.

- Next topic is how to measure the performance of our classifier.
Previously we mentioned that the optimal parameters of a model are determined (i.e., model selection) via cross-validation. For example, in the K-Nearest Neighbors algorithm, we could use the S-fold cross-validation on the training data to find the optimal value of K.
Classifier: Performance Measure

- In the K-Nearest Neighbors algorithm, for a particular value of K, we run the **S-fold cross-validation** on the training data (on all folds) and compute the **average performance measure**.
- Then, we vary K to re-run the cross-validation and compute its average performance measure.

What are the **performance measures** for a classification problem?
Classifier: Performance Measure

- Evaluating a classifier is often significantly trickier than evaluating a regressor.
- We will discuss two techniques for performance measure of classification.
  - Accuracy & Misclassification Rate / Generalization error
  - Confusion Matrix
Classifier: Performance Measure

• Accuracy & Misclassification Rate/Generalization Error:

• Accuracy is the ratio of the number of correct predictions and the total number of predictions.
Classifier: Performance Measure

• **Accuracy & Misclassification Rate/Generalization Error:**
  - Let’s look at an example of **Binary Classification**.
  - Say that we built a classifier to **predict the presence of a disease** of a set of patients.
  - There are **two possible predicted classes**: “yes” and “no”.
    - “yes”: the patient has the disease.
    - “no”: the patient doesn’t have the disease.
The classifier made a total of 165 predictions.
Out of those 165 cases, the classifier correctly predicted
- “yes” 100 times
- “no” 50 times.

Accuracy = \frac{100+50}{165} = \frac{150}{165} = 91% 

Misclassification Rate/Generalization Error = 1 – Accuracy = 9%
Classifier: Performance Measure

• **Accuracy & Misclassification Rate/Generalization Error:**

  • In some scenarios accuracy *doesn’t provide a good measure* of performance.

  • Let’s illustrate this with an example.
Classifier: Performance Measure

• Consider an example of **classifying images of handwritten digits**.
• Let’s say that we want to predict “5”s and not-“5”s.

It’s a **binary classification problem**
Assume that a classifier gives us **90% accuracy** (ratio of correct classifications over all predictions)!

How good is this accuracy measure?

Let’s say that only **10% of the images are “5”s** and **90% of the images are not-“5”s**.

Say that you wrote a **dumb** algorithm that **always predicts** that an image is **not a “5”**.

What will be its **accuracy**?

**90%!**
Classifier: Performance Measure

• Therefore, classification accuracy is generally not the preferred performance measure for classifiers.

• Especially when we are dealing with skewed datasets (i.e., when some classes are much more frequent than others).
Classifier: Performance Measure

• A much better way to evaluate the performance of a classifier is to look at the **confusion matrix**.

• The general idea is to **count** the number of times instances of **class A** are classified as **class B**.

<table>
<thead>
<tr>
<th>Actual:</th>
<th>Predicted: NO</th>
<th>Predicted: YES</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>50</td>
<td>10</td>
</tr>
<tr>
<td>YES</td>
<td>5</td>
<td>100</td>
</tr>
</tbody>
</table>

- Accuracy & Misclassification Rate /Generalization error
- Confusion Matrix
Classifier: Performance Measure

- The confusion matrix shows the ways in which our classification model is **confused** when it makes predictions.
Classifier: Performance Measure

- Let’s go back to the Binary Classification of disease example.
- Below we provide its **confusion matrix**.
- It can easily be extended to the case of **more than two classes**.

![Confusion Matrix](image)
Classifier: Performance Measure

- The number of correct and incorrect predictions are summarized with count values and broken down by each class.
- It gives us insight not only into the errors being made by our classifier.
- But more importantly the types of errors that are being made.

It is this breakdown that overcomes the limitation of using classification accuracy alone.

<table>
<thead>
<tr>
<th></th>
<th>Predicted: NO</th>
<th>Predicted: YES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual: NO</td>
<td>50</td>
<td>10</td>
</tr>
<tr>
<td>Actual: YES</td>
<td>5</td>
<td>100</td>
</tr>
</tbody>
</table>
Classifier: Performance Measure

• What can we learn from this matrix?
• There are two possible predicted classes: “yes” and “no”.
• If we were predicting the presence of a disease, for example, “yes” would mean they have the disease, and “no” would mean they don’t have the disease.

```
<table>
<thead>
<tr>
<th></th>
<th>Predicted: NO</th>
<th>Predicted: YES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NO</td>
<td>50</td>
<td>10</td>
</tr>
<tr>
<td>YES</td>
<td>5</td>
<td>100</td>
</tr>
<tr>
<td>n=165</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
Classifier: Performance Measure

• The classifier made a total of 165 predictions (e.g., 165 patients were being tested for the presence of that disease).
• Out of those 165 cases, the classifier predicted “yes” 110 times (2\textsuperscript{nd} column), and “no” 55 times (1\textsuperscript{st} column).
• In reality, 105 patients in the sample have the disease (2\textsuperscript{nd} row), and 60 patients do not (1\textsuperscript{st} row).

<table>
<thead>
<tr>
<th></th>
<th>Predicted: NO</th>
<th>Predicted: YES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual: NO</td>
<td>50</td>
<td>10</td>
</tr>
<tr>
<td>Actual: YES</td>
<td>5</td>
<td>100</td>
</tr>
</tbody>
</table>

We will define some terms to understand the type of errors.
Let’s now define the most **basic terms**.

**True positives (TP):** These are cases in which we predicted yes (i.e., patients have the disease), and they **actually have** the disease.

**True negatives (TN):** We predicted no, and they **don’t have** the disease.

These are whole numbers (**not rates**).
 Classifier: Performance Measure

- **False positives (FP):** We predicted yes, but they don’t actually have the disease. (Also known as “**Type I error**”)
- **False negatives (FN):** We predicted no, but they actually do have the disease. (Also known as “**Type II error**”)

<table>
<thead>
<tr>
<th></th>
<th>Predicted: NO</th>
<th>Predicted: YES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual: NO</td>
<td>TN = 50</td>
<td>FP = 10</td>
</tr>
<tr>
<td>Actual: YES</td>
<td>FN = 5</td>
<td>TP = 100</td>
</tr>
</tbody>
</table>

**FP:** Incorrect prediction - Patient doesn’t have cancer, yet you predicted it.

**FN:** Patient has cancer, but you missed it!
Classifier: Performance Measure

- A perfect classifier would have only **true positives (TP)** and **true negatives (TN)**.
- So its confusion matrix would have **nonzero values only on its main diagonal** (top left to bottom right).

<table>
<thead>
<tr>
<th>Actual: NO</th>
<th>Predicted: NO</th>
<th>Predicted: YES</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TN = 50</strong></td>
<td><strong>FP = 10</strong></td>
<td><strong>60</strong></td>
</tr>
<tr>
<td><strong>FN = 5</strong></td>
<td><strong>TP = 100</strong></td>
<td><strong>105</strong></td>
</tr>
<tr>
<td><strong>55</strong></td>
<td><strong>110</strong></td>
<td></td>
</tr>
</tbody>
</table>

**Perfect Classifier:**
No wrong prediction & no miss!
Classifier: Performance Measure

- The confusion matrix gives us a lot of information.
- But sometimes we may prefer a more concise metric.
- An interesting one to look at is the accuracy of the positive predictions.
- It is called the precision of the classifier.

<table>
<thead>
<tr>
<th></th>
<th>Predicted: NO</th>
<th>Predicted: YES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual: NO</td>
<td>TN = 50</td>
<td>FP = 10</td>
</tr>
<tr>
<td>Actual: YES</td>
<td>FN = 5</td>
<td>TP = 100</td>
</tr>
</tbody>
</table>

\[
\text{precision} = \frac{TP}{TP + FP}
\]

**Precision:** I care only about my predictions (ignore the missed ones)
Classifier: Performance Measure

- A trivial way to have **perfect precision** is to make one single positive prediction and ensure it is correct (\(\text{precision} = \frac{1}{1} = 100\%\)).
- What is the precision below?
  - \(\frac{TP}{TP + FP} = \frac{3}{3} = 100\%\)
  - But this is **not very useful** as it missed some “5”s!

Classifier that predicts “5” and “not-5”s

\[
\text{precision} = \frac{TP}{TP + FP}
\]
So, precision is typically used along with another metric named **recall**, also called **sensitivity** or **true positive rate (TPR)**.

- It’s a measure of the **missed predictions**.
- Recall is the ratio of positive instances that are correctly detected by the classifier.

\[
\text{precision} = \frac{TP}{TP + FP}
\]

\[
\text{recall} = \frac{TP}{TP + FN}
\]

Precision = \(\frac{3}{3+0} = 100\%\)

Recall = \(\frac{3}{3+2} = 60\%\)
Classifier: Performance Measure

- It is often convenient to combine precision and recall into a single metric called the $F_1$ score.
- The $F_1$ score is the harmonic mean of precision and recall.

$$F_1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} = \frac{2}{\frac{1}{\text{precision}} + \frac{1}{\text{recall}}} = \frac{TP}{TP + \frac{FN + FP}{2}}$$

It provides a simple way to compare two classifiers.
Classifier: Performance Measure

- Recall the regular mean treats *all values equally*.
- However, the harmonic mean *gives much more weight to low values*.
- As a result, the classifier will only get a **high $F_1$** score if *both* recall and precision are high.

$$F_1 = \frac{2 \cdot \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} = \frac{TP}{TP + \frac{FN + FP}{2}}$$
Classifier: Performance Measure

• The F₁ score **favors classifiers** that have similar precision and recall.

• This is **not always** **what we want**.

• In some contexts we mostly care about precision, and in other contexts we really care about recall.

• Let’s explain this using two scenarios.
**Classifier: Performance Measure**

- **Scenario 1**: Consider a classifier to detect videos that are **safe (good) for kids**.
- We would prefer a classifier that **wouldn’t tag a harmful video as good** (i.e., the 1st quadrant should be 0)!
- So, it may reject many good videos (**low recall**) but keeps only safe ones (**high precision**).

We don’t want a classifier that has a much higher recall but lets a few really **harmful videos** show up in our product.
Classifier: Performance Measure

- **Scenario 2**: Suppose we train a classifier to detect shoplifters on surveillance images.
- We prefer a classifier that *won’t miss a shoplifter* (i.e., the 3rd quadrant should be 0).
- It is probably fine if our classifier has only 25~30% precision as long as it has **99~100% recall**.
- The security guards will get a few false alerts, but almost **all shoplifters will get caught**.
Classifier: Performance Measure

- Unfortunately, **we can’t have it both ways**!
- Increasing precision reduces recall, and vice versa.
- This is called the **precision/recall tradeoff**.

Classifier that predicts “5” and “not-5”s
Classifier: Performance Measure

• Suppose in the handwritten digit (5 vs. not 5) example, the decision threshold is positioned at the central arrow (between the two 5s).
• We will find 4 true positives (actual 5s) on the right of that threshold, and one false positive (actually a 6).
• Therefore, with that threshold, the precision is 80% (4 out of 5).
• But out of 6 actual 5s, the classifier only detects 4, so the recall is 67% (4 out of 6).
Classifier: Performance Measure

• Now let’s **raise the threshold** (move it to the arrow on the right).
• The false positive (the 6) becomes a true negative, thereby **increasing precision** (up to 100% in this case).
• But one true positive becomes a false negative, **decreasing recall down to 50%**.
• Conversely, lowering the threshold increases recall and reduces precision.

![Diagram showing precision and recall with various thresholds and scores](image)
Classifier: Performance Measure

• So how can we decide \textbf{which threshold to use}?  
• We can plot precision and recall as \textbf{functions of the threshold value}.

![Graph showing Precision-Recall Curve](chart.png)

\textit{It’s called the Precision-Recall Curve (assignment 1)}
Another way to select a good precision/recall tradeoff is to plot precision directly against recall.

See that precision really starts to fall sharply around 80% recall.

We will probably want to select a precision/recall tradeoff just before that drop, for example, at around 60% recall.

But of course the choice depends on the project.
Classifier: Performance Measure

- Another common tool used with **binary classifiers** is the **Receiver Operating Characteristic (ROC)** curve.
- It is very similar to the precision/recall curve.

\[
\text{recall} = \frac{TP}{TP + FN}
\]

The ROC curve plots the **true positive rate** (another name for **recall**) against the **false positive rate**.
Performance Measure: ROC Curve

- Let’s understand the concept of true positive rate (TPR) and the false positive rate (FPR).
- TPR is a **normalized measure of true positives**.
- TPR is the ratio of the true positive predictions (# Predicted YES True) and the ground truth of positive instances (#Actual YES).

\[
\text{TPR} = \frac{\text{# Predicted YES (True)}}{\text{# Actual Yes}}
\]

\[
\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}}
\]

For a good classifier, its TPR should be large (ideally 1)
Performance Measure: ROC Curve

- FPR is a **normalized measure of false positives**.
- No. of negative instances that **inaccurately predicted to be positive**.
- FPR is the **ratio of the false positive** predictions (# Predicted YES False) and the **ground truth of negative instances** (#Actual NO).

<table>
<thead>
<tr>
<th></th>
<th>Predicted: NO</th>
<th>Predicted: YES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual: NO</td>
<td>TN = 50</td>
<td>FP = 10</td>
</tr>
<tr>
<td>Actual: YES</td>
<td>FN = 5</td>
<td>TP = 100</td>
</tr>
</tbody>
</table>

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}$$

For a good classifier, its **FPR should be small** (ideally 0)
**TPR:** What % of patients (positive instances) did we **correctly identify**?

**FPR:** How frequently did we “hallucinate” about detecting the disease?

TPR = $\frac{\text{# Predicted YES (True)}}{\text{# Actual Yes}}$

TPR = $\frac{TP}{TP+FN}$

For a good classifier, its **TPR** should be large (ideally 1)

FPR = $\frac{\text{# Predicted YES (False)}}{\text{# Actual NO}}$

FPR = $\frac{FP}{FP+TN}$

For a good classifier, its **FPR** should be small (ideally 0)

<table>
<thead>
<tr>
<th>n=165</th>
<th>Predicted: NO</th>
<th>Predicted: YES</th>
<th>TPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual: NO</td>
<td>TN = 50</td>
<td>FP = 10</td>
<td>60</td>
</tr>
<tr>
<td>Actual: YES</td>
<td>FN = 5</td>
<td>TP = 100</td>
<td>105</td>
</tr>
</tbody>
</table>

| | 55 | 110 | **FPR:** How frequently did we “hallucinate” about detecting the disease? |
Performance Measure: ROC Curve

- Let’s explain the **usefulness** of TPR and FPR.
- Say that we created 3 **dumb classifiers**.
- **Classifier 1**: predicts everybody has the disease
- **Classifier 2**: predicts nobody has the disease
- **Classifier 3**: predicts the disease based on a fair coin toss (i.e., 50% has the disease)

\[
\begin{align*}
TPR &= \frac{TP}{TP+FN} \\
FPR &= \frac{FP}{FP+TN}
\end{align*}
\]

**Classifier 1**

\[
\begin{align*}
TPR &= \frac{105}{105} = 1 \\
FPR &= \frac{60}{60} = 1
\end{align*}
\]

**Ideal TPR**

**Worst FPR**

**Predicts only YES**

<table>
<thead>
<tr>
<th>n=165</th>
<th>Predicted: NO</th>
<th>Predicted: YES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual: NO</td>
<td>TN = 0</td>
<td>FP = 60</td>
</tr>
<tr>
<td>Actual: YES</td>
<td>FN = 0</td>
<td>TP = 105</td>
</tr>
<tr>
<td></td>
<td>55</td>
<td>110</td>
</tr>
</tbody>
</table>
Performance Measure: ROC Curve

- **Classifier 1**: predicts everybody has the disease
- **Classifier 2**: predicts nobody has the disease
- **Classifier 3**: predicts the disease based on fair coin toss (i.e., 50% has the disease)

**Classification Table**

<table>
<thead>
<tr>
<th>Actual:</th>
<th>Predicted:</th>
<th>Predicted:</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>NO</td>
<td>YES</td>
</tr>
<tr>
<td></td>
<td>TN = 60</td>
<td>FP = 0</td>
</tr>
<tr>
<td>YES</td>
<td>FN = 105</td>
<td>TP = 0</td>
</tr>
</tbody>
</table>

- FPR = \(\frac{0}{60} = 0\)
- TPR = \(\frac{0}{105} = 0\)

**Performance Measures**

- TPR = \(\frac{TP}{TP+FN}\)
- FPR = \(\frac{FP}{FP+TN}\)
Performance Measure: ROC Curve

- **Classifier 1**: predicts everybody has the disease
- **Classifier 2**: predicts nobody has the disease
- **Classifier 3**: predicts the disease based on fair coin toss (i.e., 50% has the disease)

### ROC Curve Formulas

\[
\text{TPR} = \frac{TP}{TP + FN}
\]

\[
\text{FPR} = \frac{FP}{FP + TN}
\]

#### Classifier 3

- TPR = \(\frac{52}{105} \approx 0.5\)
- FPR = \(\frac{30}{60} = 0.5\)

#### Predicts by a Fair Coin Toss

<table>
<thead>
<tr>
<th>Actual:</th>
<th>Predicted:</th>
<th>Predicted:</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>NO</td>
<td>YES</td>
</tr>
<tr>
<td>TN = 30</td>
<td>FP = 30</td>
<td>30</td>
</tr>
<tr>
<td>FN = 53</td>
<td>TP = 52</td>
<td>60</td>
</tr>
<tr>
<td>55</td>
<td>110</td>
<td>105</td>
</tr>
</tbody>
</table>

n=165
Performance Measure: ROC Curve

**Classifier 1**: All predictions are correct (**no mistakes**), but suffers from **worst hallucinations**!

**Classifier 2**: No hallucinations, **worst predictions**!

**Good classifier**: optimizes both TPR (to 1) and FPR (to 0).

**Classifier 1**

- TPR = \( \frac{105}{105} = 1 \)  
  - Ideal TPR
- FPR = \( \frac{60}{60} = 1 \)  
  - Worst FPR

**Classifier 2**

- TPR = \( \frac{0}{105} = 0 \)  
  - Worst TPR
- FPR = \( \frac{0}{60} = 0 \)  
  - Ideal FPR
Performance Measure: ROC Curve

- The **optimal TPR and FPR** depend on the **threshold** for classification.
- To determine the optimal threshold (**optimal TPR and FPR**) we generate multiple TPR-FPR pairs by **varying the threshold** and plot a graph.
- It’s known as the **Receiver Operating Characteristic (ROC)** curve.

Let’s understand the ROC curve.
Performance Measure: ROC Curve

Ideal Classifier: (TPR = 1, FPR = 0)

Classifier 1: Everybody has disease (TPR = 1, FPR = 1)

Classifier 2: Nobody has disease (TPR = 0, FPR = 0)

Classifier 3: Random Guess (TPR = 0.5, FPR = 0.5)
We report results of our binary classification by plotting the ROC curve (blue in the figure) in the TPR-FPR space.

The goal of the ROC curve is to identify an optimal threshold for effective predictions.

This method was developed during World War II to detect Japanese aircrafts.

How do we generate this graph for K-NN to identify the optimal threshold?
Performance Measure: ROC Curve

- We predict the class $y$ of a test point $\mathbf{x}$ by using a threshold $t$:
  $$\hat{y}(\mathbf{x}) = \text{sign}[\text{probability of } \mathbf{x} \geq t]$$
- For example, when $t = 0.5$, the probability of the two test points $\mathbf{a}$ and $\mathbf{b}$ to belong to “red” class are 1 and 0.6.

Say we set threshold at $t = 0.7$

- $\hat{y}(\mathbf{a}) = \text{sign}[\text{probability of } \mathbf{a} \geq 0.7]$
  $$= \text{sign}[1 \geq 0.7] = + \text{ (red circle)}$$

- $\hat{y}(\mathbf{b}) = \text{sign}[\text{probability of } \mathbf{b} \geq 0.7]$
  $$= \text{sign}[0.6 \geq 0.7] = - \text{ (yellow smiley)}$$

$\mathbf{a}$: $p(y = \text{red} | \mathbf{a}, D, K = 5) = 5/5 = 1$

$\mathbf{b}$: $p(y = \text{red} | \mathbf{b}, D, K = 5) = 3/5 = 0.6$
Performance Measure: ROC Curve

- **By varying the threshold $t$ from 0 to 1**, we get the ROC curve.
- At $t = 1$, $\hat{y}(\vec{x}) = 0 \ \forall \vec{x} \Rightarrow$ Nobody has the disease (Classifier 2)
- At $t = 0$, $\hat{y}(\vec{x}) = 1 \ \forall \vec{x} \Rightarrow$ Everybody has the disease (Classifier 1)
Performance Measure: ROC Curve

• The dotted line represents the ROC curve of a purely random classifier.

• Also, note that the higher the TPR (recall), the more false positives (FPR) the classifier produces (more hallucination).

Thus, there is a tradeoff!

A good classifier stays as far away from that line as possible (toward the top-left corner).
Performance Measure: ROC Curve

• For our classification problem, we generate the ROC curve by varying threshold and see how close it is to the ideal classifier.
• We report the ROC curve as a performance measure of our classifier.

A good classifier stays as far away from that line as possible (toward the top-left corner).
Performance Measure: ROC Curve

- How do we compare two binary classifiers **numerically** using their ROC curves?
- We can measure the **area under the curve (AUC)**.

A perfect classifier will have a ROC AUC equal to 1.

Whereas a purely random classifier will have a ROC AUC equal to 0.5.
Before we end the discussion on various performance measures, we want to define accuracy of a classifier in terms the metrics of the confusion matrix.

Accuracy is the ratio of the number of correct predictions and the total number of predictions.

Accuracy = \( \frac{TP + TN}{n} \)

\[
\begin{align*}
TPR &= \frac{TP}{TP + FN} \\
FPR &= \frac{FP}{FP + TN}
\end{align*}
\]

<table>
<thead>
<tr>
<th>n=165</th>
<th>Predicted: NO</th>
<th>Predicted: YES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual: NO</td>
<td>TN = 50</td>
<td>FP = 10</td>
</tr>
<tr>
<td>Actual: YES</td>
<td>FN = 5</td>
<td>TP = 100</td>
</tr>
<tr>
<td></td>
<td>55</td>
<td>110</td>
</tr>
</tbody>
</table>

Misclassification Rate/Generalization Error = 1 – Accuracy = 0.09
Confidence on the Performance Measure

How much confidence do we have about an estimation?
Confidence about an Estimation

• Let’s say that we have computed the **accuracy of a classifier**.

• How much **confidence** do we have on the estimated accuracy?

• In other words, how do we measure the **uncertainty** of a **proportion statistic** (e.g., accuracy or error).

• This is measured by the **confidence interval**.
Confidence Interval

- A confidence interval (CI) is a range of scores **likely to contain the parameter being estimated**.
- For example, a political poll might indicate that a candidate is **likely to get 52%** of the vote +/-4% with a **confidence level** of 95%.
- I.e., the confidence interval is of size 8.
- It means that the pollster believes that **95% of the time** the candidate will receive between **48% and 56% of the vote**.

A **wider confidence interval** indicates **larger uncertainty** in the prediction.
Confidence Interval

- The standard is to compute the CI at the 95% confidence level by finding the lower & upper bound.

- Example: 95% confidence level means that if we repeat an experiment over and over again, 95 percent of the time our results will match the results we get from a population.
Confidence Interval

• Together the confidence interval and the confidence level are intended to indicate the reliability of the estimate.

• Almost always, increasing the confidence level will require widening the confidence interval.
How do we compute the confidence interval of a K-NN performance measure (e.g., accuracy)?

First, we need to specify at what confidence level we want to measure the confidence interval. Generally we use 95% confidence level for measuring the confidence interval.

Then, we compute the lower & upper bounds of the estimate.
Confidence Interval: K-NN

• Formally, at the 95% confidence level, the confidence interval for a proportion statistic \( p \) (e.g., accuracy) is given by (for \( n \) samples):

\[
\text{Lower bound} = p - z \times \text{standard error}
\]

\[
\text{Upper bound} = p + z \times \text{standard error}
\]

\[
\text{standard error} = \sqrt{\frac{\text{variance}}{n}} = \sqrt{\frac{p(1 - p)}{n}}
\]

\( z = 1.96 \) for 95% confidence level

Binary classification is modeled by the Bernoulli distribution which has the variance: \( p(1 - p) \)
Confidence Interval

• To **further understand** the concept of confidence level and interval, and the formula for CI, you need to **read through the slides on confidence interval**.

• It’s **very important** that you understand the concepts and the formula.