**1/ Orthogonalization:**

* Fit training set well on cost function => bigger network, Adam…
* Fit dev set well on cost function => Regularization, Bigger Training Set…
* Fit test set well on cost function => Bigger Dev Set…
* Performs well in real world => Change dev set distribution or cost function

**2/ Set up your goal:**

a/ Single number evaluation metric:

Diagram

Description automatically generated with medium confidence

* Precision: What % that your classify as cat and what % are cats
* Recall: What % that your classfier recognize as cat that actually cat
* Problem: if classifier A do better on Recall and B do better on Precision then you don’t know which model perform better
* Instead do the F1 Score = a combination of Precision and Recall. So having 1 Dev set and a single number of real evaluation metric can allow to know which classifier is better

Table

Description automatically generated

b/ Satisficing and optimizing metrics:

A picture containing chart

Description automatically generated

c/ Train/Dev/Test Distribution:

* Test and Dev set should come from the same distribution

d/ Size of dev and test sets:

A picture containing chart

Description automatically generated- The size of test set = set your test set to be big enough to give high confidence in the overall performance of your systems

e/ In case you have the target wrong, know when to change the dev/test sets and metrics:

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, timeline

Description automatically generated

* If B performed better on User images then pick B or change your dev/test and re evaluate

**3/ Comparing to Human-level performance:**

* You can’t surpass Bayes Optimal Error

Diagram

Description automatically generated with medium confidenceGraphical user interface, text, application

Description automatically generated

a/ Avoidable bias:

* Solution: Bigger NN,…

Graphical user interface, application, Word

Description automatically generated

* Depends on how the training/dev error compare to human’s error, we might want to focus on reducing bias or variance
* 8% error is high compared to 1% => bias.
* Variance: Regularizaton or get more training data => Reduce Dev Error to training error.

b/ Define human-level performance:

Table

Description automatically generated

A picture containing diagram

Description automatically generated

Diagram

Description automatically generated

c/ Surpassing human-level performance:

A picture containing text, person, screenshot

Description automatically generated

* Bayes error < 0.5%. Avoidable bias is 0.1 since we ignore 1% of 1 person. Variance = 0.2%. So the goal of improving variance is better than the bias one

Graphical user interface, text, application

Description automatically generated

d/ Improving your model performance:

* Two fundamental assumptions of SL:

+ You can fit the training set pretty well => You have avoided bias

+ The training set performance generalized pretty well to the dev/test set => solved variance

Diagram

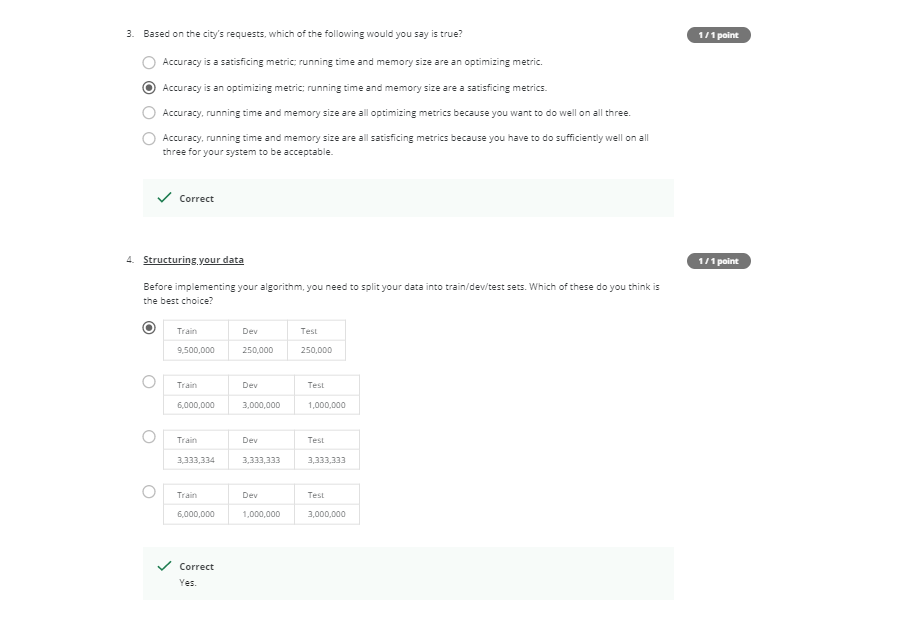
Description automatically generated

Graphical user interface, website

Description automatically generated

Graphical user interface, application

Description automatically generated



Graphical user interface, text, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, application, Teams

Description automatically generated

Table

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Text

Description automatically generated

Graphical user interface, application, Teams

Description automatically generated