Mudcard

- **can this method be used to explain outliers?**
  - any point in a supervised ML problem can be explained by shap
- **For a linear model, will the shap values be the same for each data point? Since it is determined by the coefficients of the equation.**
  - nope
  - the coefficients are the same for each feature, but the feature values change for each point and so does the shap values
- **Is SHAP appropriate for multi-class classification problems? Or is it too computationally expensive?**
  - it works with multi-class problems
  - I wouldn't exclude a method just because it is computationally expesive
- **Are you basically saying that mean(|SHAP value|) is another metric to measure global importance?**
  - yes! :)
- **I have difficulty understanding the plots of feature value vs. shap value. How should we interpret them?**
  - play around with the code, make sure you understand every line
  - read the linked papers and books
- **Do we have a general sheet for the formulas that we have learnt in this course for the final exam?**
  - not really
  - it's an open-book exam so I won't ask questions you can simply copy-paste from a sheet or the lecture notes
  - the exam tests your understanding of ML concepts

A note on imbalanced datasets

- we learnt that a classification problem is imbalanced if more than 90-95% of the points belong to one class (class 0) and only a small fraction of the points belong to the other class (class 1)
  - fraud detection
  - sick or not sick (usually by far most people are not sick)
- we learnt to not use a metric that relies on the True Negatives in the confusion matrix
  - no accuracy or ROC
  - use f_beta or the precision-recall curve instead

What else can I do if I have an imbalanced dataset?

- most (but not all) classification algorithms we covered have a parameter called `class_weight` which allows you to assign more weight to the class 1 point
  - a misclassified class 1 point will contribute more to the cost function than a misclassified class 0 point
  - read the manual on `class_weight` because the different algorithms have slightly different definitions for this parameter
  - usually you can use `None`, `balanced`, or manually define what the class weight should be
  - it is worthwhile to tune this parameter if you have an imbalanced dataset
- resample/augment the dataset
  - SMOTE (Synthetic Minority Over-sampling Technique), see the [paper]
  - to improve the balance of the problem, new class 1 examples are synthesized from the existing examples
  - be careful though!
    - while resampling improves the balance of the dataset, the results of the model can be misleading
    - when you deploy the model, the incoming data will be as imbalanced as the original data

Misleading results with resampling

- let's assume you have an imbalanced dataset with 99% of points in class 0 and 1% of points in class 1
- you resample it such that the improved class balance is 50-50
- here the confusion matrix of the trained model:

		Predicted class	
		Predicted Negative (0)	Predicted Positive (1)
Actual class	Condition Negative (0)	**True Negative (TN): 45%**	**False Positive (FP): 5%**
	Condition Positive (1)	**False Negative (FN): 5%**	**True Positive (TP): 45%**

- 90% accuracy which is well above the 50% baseline accuracy!
- the precision, recall, and f1 scores are all 0.9.
- it looks great, doesn't it?
- let's rewrite the confusion matrix to reflect rates with respect to the Condition Negative and Condition Positive points!

		Predicted class	
		Predicted Negative (0)	Predicted Positive (1)
Actual class	Condition Negative (0) 50% of the points	**90% of CNs are correctly classified**	**10% of CNs are incorrectly classified**
	Condition Positive (1) 50% of the points	**10% of CPs are incorrectly classified**	**90% of CPs are correctly classified**

Let's deploy this model

- the incoming data has the same balance as the original dataset (99% to 1%)
- let's assume we have 1e5 new points, 9.9e4 belongs to class 0, 1000 belongs to class 1
- what will be the numbers in the confusion matrix?

		Predicted class	
		Predicted Negative (0)	Predicted Positive (1)
Actual class	Condition Negative (0) 99000 points	**True Negative (TN): 99000 * 0.9 = 89100**	**False Positive (FP): 99000 * 0.1 = 9900**
	Condition Positive (1) 1000 points	**False Negative (FN): 1000 * 0.1 = 100**	**True Positive (TP): 1000 * 0.9 = 900**

- the accuracy of this model is still 0.90 but now it is well below the baseline of 0.99!
- recall is good (0.90) but the precision is not great (~0.083)
- the f1 score is ~0.15
- the false positives are overwhelming
- this is why you need to be careful with resampling

Quiz

Mudcard