Conformal Prediction and Applications

Adithya K Anil **Rolla Siddharth Reddy** **Nikhil Jamuda** **Pasupuleti Dhruv Shivkant**

Abstract

Conformal Prediction is a statistical framework that provides rigorous, finite-sample guarantees for the predictions made by machine learning models. Unlike traditional predictors that produce a single-point estimate, conformal predictors generate prediction sets, ensuring a user-specified coverage probability under minimal assumptions. The strength of this methodology lies in the minimal set of assumptions it requires. This produces coverage guarantees without any assumptions on the accuracy or calibration of the underlying model, even in non-asymptotic regimes. Thus it provides a robust quantification of uncertainty where it is very essential. In this project, we aim to empirically evaluate and reproduce results to demonstrate this coverage using multiple configurations and algorithms designed for conformal predictions and test it in various practical settings.

1 Introduction

Conformal prediction provides a framework for constructing set-valued predictors with formal guarantees of their reliability. Given a trained predictive model, conformal methods are constructed around it to generate prediction sets that, with high probability, contain the true label. The central guarantee is *marginal coverage*, i.e., the prediction set contains the true label with probability at least $1 - \alpha$, averaged over the test set distribution. Mathematically this is equivalent to, $\mathbb{P}(Y_{\text{test}} \in C(X_{\text{test}})) \geq 1 - \alpha$. This guarantee is model-agnostic and distribution-free under the assumption that the data points are exchangeable. While trivial prediction strategies (e.g., outputting the full label set with probability $1 - \alpha$, and the empty set otherwise) satisfy the marginal coverage property, they are uninformative. The goal is instead to construct *compact prediction sets* that reflect the model's confidence and adapt to uncertainty in the input. There are two main variants of conformal prediction, **Split Conformal Prediction** [4.1] and **Full Conformal Prediction** [4.2]. Conformal methods apply to both classification and regression. Importantly, we also seek *adaptive* prediction sets—small for "easy" samples and larger for "hard" ones. This adaptivity is one of the central goals in conformal prediction. A stronger form—**Adaptive prediction sets** and **Regularized Adaptive prediction sets (RAPS)** [4.4]—are methods introduced to tackle this problem and achieve better guarantees. In this report, we reproduce key empirical results demonstrating the coverage guarantee of conformal predictors. We further explore how design choices such as score function selection and calibration size influence prediction set size and coverage performance.

2 Methodology

Conformal prediction is a distribution-free framework for uncertainty quantification in machine learning. Given a trained model and a desired confidence level $\alpha \in (0, 1)$, it constructs a prediction set $C(x)$ such that the *marginal coverage* guarantee holds, i.e. for a freshly sampled (X_{test}, y_{test}), $P(y_{test} \in C(X_{test}) \geq 1 - \alpha$.

Theorem 1 (Main Theorem)**.** *Let $\{(X_i, Y_i)\}_{i=1}^n$ be exchangeable samplings and let s be any valid (i.e., symmetric) nonconformity score. Then, for any new point (X_{n+1}, Y_{n+1}) satisfying the exchangeability criterion, we have*

$$\mathbb{P}(Y_{n+1} \in C(X_{n+1})) \geq 1 - \alpha,$$

where $C(x) = \{y : s(x, y) \leq \hat{q}\}$, and $\hat{q}$ is the $\frac{\lceil (1-\alpha)(1+n) \rceil}{n}$ empirical quantile from the calibration scores.

For example, in **RAPS** the score function is

$$s_{\text{RAPS}}(x, y) := \Big\{ \underbrace{\rho_x(y)}_{\text{cumulative softmax sum}} + \underbrace{\hat{\pi}_x(y) \cdot u}_{\text{randomized term}} + \underbrace{\lambda \cdot |(o_x(y) - k_{reg})|}_{\text{regularization term}} \Big\}$$

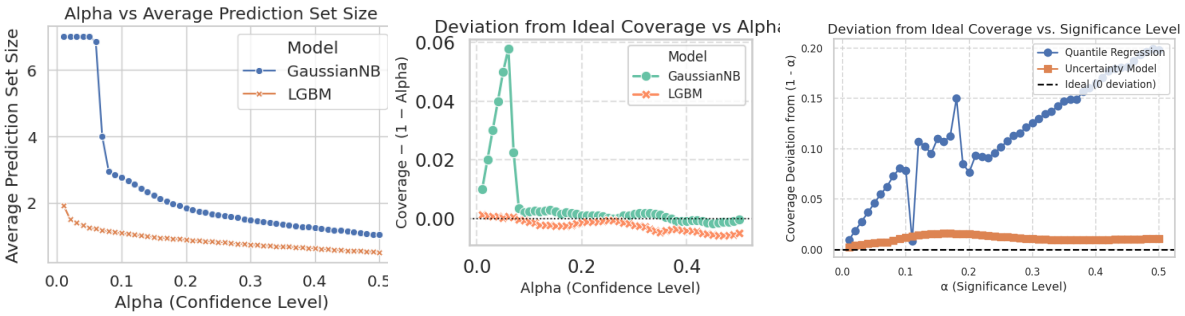

Figure 1: Split conformal prediction methods analysis done using 2 models. Refer [4.1]

This algorithm [2] is a general form of conformal prediction. It uses a pre-trained predictor model f and a held-out calibration set. For each calibration point, it computes a nonconformity score and sorts these scores in ascending order. It then determines a threshold score using a quantile based on the desired confidence level α. The prediction set for a test point is constructed by including all labels whose scores fall below this threshold. The score function here represents any arbitrary negatively oriented score function that can be used as long as it is symmetric over the data points.

Algorithm 1 Conformal Prediction with Score Function $s(x, y)$

Require: Model f, calibration set $\{(x_i, y_i)\}_{i=1}^{n}$, test input x, confidence level α

for all (x_i, y_i) in calibration set **do**
  Compute $s_i = s(x_i, y_i)$
Sort scores $\{s_1, \ldots, s_n\}$
$\hat{q} \leftarrow \text{Quantile}_{\lceil (n+1)(1-\alpha) \rceil / n}$
$C(x) \leftarrow \emptyset$
for all label y in output space **do**
  if $s(x, y) \leq \hat{q}$ **then**
    Add y to $C(x)$
return $C(x)$

3 Experiments

We test the various methods of conformal prediction by computing metrics such as marginal coverage, average prediction set sizes, etc., to test the procedures for empirical verification. For general split conformal prediction using multiple models, the results can be seen in Fig [1] and [2].

3.1 Split Conformal Prediction Analysis under different settings

We simulate multiple models using the split conformal method and then calculate the variation of marginal coverages and their deviation from the ideal values for multiple confidence values. Refer [1].

3.2 Comparison of quantile regression v/s uncertainty prediction method

We have implemented 2 different methods(Quantile Regression and Conformalized uncertainty predictors) for conformal prediction on regression models. From theory, we know that Quantile Regression is a more robust method because of its direct dependence on the significance level α in contrast to Conformalized uncertainty

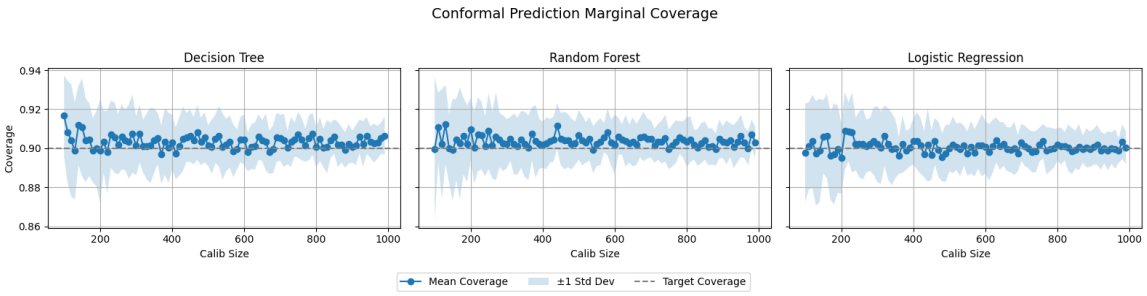

Figure 2: Marginal coverage with deviation bands - random sampling 500 times per n_{cal}

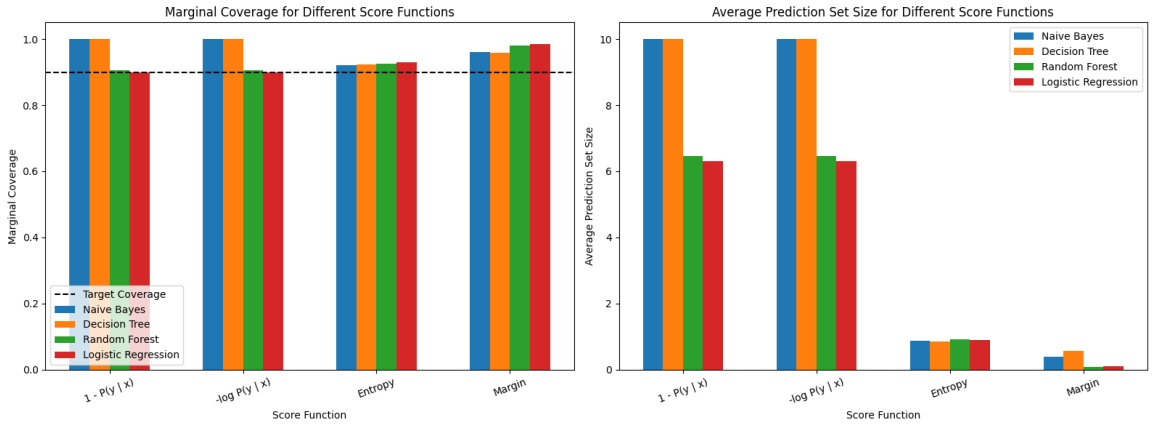

Figure 3: Comparison of score functions. *Additional Info:* The graphs of density vs Marginal Coverage for different values of calibration data set size can be seen in Fig [10]

predictor methods. By doing the experiment, we try to empirically confirm this result, and the results can be seen in Fig [1]

3.3 Analysis on score function and calibration sets

We analyze different kinds of score functions that could have been used in this procedure. These are all symmetric score functions, and we test the coverages and average prediction set sizes using them. More details are in the Appendix. Results in Fig [3] and [10].

3.4 Regularized Adaptive prediction sets - (RAPS)

In this experiment, we try to recreate the results from the paper [1] using the Imagenet Dataset

3.4.1 RAPS vs Other Methods

We implemented the RAPS algorithm using the CNN model Resnet152[11] on ImageNet-Val(by UCB)(achieving around 76% accuracy) and compared it with other methods for multiple values of α. We have achieved that RAPS has reduced size significantly compared to other methods while achieving the desired coverage. The results can be seen in Fig [5]

True: capuchin, ringtail, Cebus capucinus
Set: capuchin (0.642), guenon (0.745), titi (0.842), macaque (0.889)

True: sundial
Set: sundial (0.841), schooner (0.870), pirate ship (0.884), clock (0.897), telescope (0.915)

True: cardoon
Set: cardoon (0.853), bee (0.881), artichoke (0.908)

Figure 4: Prediction sets with confidence scores. Each set includes the true class along with visually similar alternatives and their corresponding prediction scores.

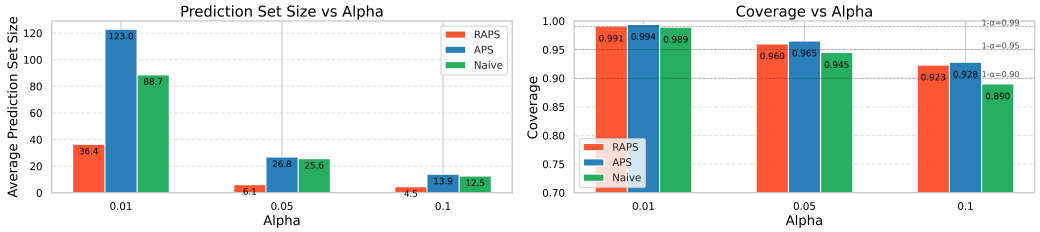

Figure 5: RAPS vs Other Methods on ResNet152 with ImageNet-Val (by UCB).

3.4.2 Finding Optimal k_{reg} and λ for RAPS and Comparing on different dataset and models

Here we find the optimal k_{reg} by Algorithm[4] for each value of α. By doing a simple grid search over different values of λ(Additional examples in [3]), we find $\lambda_{optimal} = 0.1$. We take these optimal values and compare different models for $\alpha = 0.1$ in Table 1

Alpha	**RAPS**	
	Size	Coverage
0.01	**34.825** | 36.408	**0.991** | 0.991
0.05	**6.004** | 6.055	**0.958** | 0.960
0.10	**3.993** | 4.510	**0.930** | 0.933

(a) k_{reg} RAPS

Model	Coverage			Size	
	APS	RAPS	Δ	APS	RAPS
ResNet18	0.933	**0.903**	0.003	19.177	**4.392**
ResNet50	0.943	**0.911**	0.011	15.230	**2.644**
ResNet101	0.942	**0.909**	0.009	12.546	**2.281**
ResNet152	0.942	**0.909**	0.008	11.855	**2.137**
ResNeXt101	0.937	**0.901**	0.001	17.127	**1.401**
VGG16_BN	0.938	**0.909**	0.009	14.423	**3.235**

(b) Comparison of models on ImageNet-Val

Table 1: (a) Shows how optimal k_{reg} for different values of α affect prediction size and coverage. Bold values are **with** optimal k_{reg} and rest is without optimal k_{reg}(b) Shows the comparison of RAPS with different models with $\alpha = 0.1$. Δ is the deviation from the similar experiment in the paper [1].

3.4.3 Set size comparison in RAPS

We compare the number of samples that fall under each of the rank (quantile) sections of the dataset. This gives a rough idea of the hardness distribution of the data points. We clearly see that a fair ratio of samples lie in the lower ranks of their predictions which is handled by RAPS. Results are seen in [2].

Rank range	No. of samples	Avg. set size
≤ 5	9011	4.723
5 - 10	387	13.778
10 - 30	338	16.716
30 - 60	129	19.860
60 - 1000	135	23.029

Table 2: Average set sizes on ImageNet-V2 using VGG16 model for various rank ranges.

4 Conclusion

In this project, we explored the general framework of conformal prediction along with some improved models to achieve efficient and adaptive prediction sets. We tested some hypothesis and design choices of this procedure empirically and verified them. The experiments demonstrate trade-offs and dependencies on certain parameters and how they affect the final coverage.

References

[1] Anastasios N Angelopoulos, Stephen Bates, Jitendra Malik, and Michael I Jordan. Uncertainty sets for image classifiers using conformal prediction. *arXiv preprint arXiv:2009.14193*, 2021.

[2] Rina Foygel Barber, Emmanuel J. Candes, Aaditya Ramdas, and Ryan J. Tibshirani. Predictive inference with the jackknife+, 2020.

[3] S. Bates, E. Candès, L. Lei, Y. Romano, and M. Sesia. Testing for outliers with conformal p-values. *arXiv:2104.08279*, 2021.

[4] Jock Blackard. Covertype. UCI Machine Learning Repository, 1998. DOI: https://doi.org/10.24432/C50K5N.

[5] Aaditya K. Ramdas Chirag Gupta, Arun K. Kuchibhotla. Nested conformal prediction and quantile out-of-bag ensemble methods, 2022.

[6] Yu Sun Kilian Q. Weinberger Chuan Guo, Geoff Pleiss. On calibration of modern neural networks, 2017.

[7] Thibault Cordier, Vincent Blot, Louis Lacombe, Thomas Morzadec, Arnaud Capitaine, and Nicolas Brunel. Flexible and Systematic Uncertainty Estimation with Conformal Prediction via the MAPIE library. In *Conformal and Probabilistic Prediction with Applications*, 2023.

[8] L. Guan and R. Tibshirani. Prediction and outlier detection in classification problems. *arXiv:1905.04396*, 2019.

[9] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. 2017. arXiv:1706.04599.

[10] Stefan Güttel, Yuji Nakatsukasa, Marcus Webb, and Alban Bloor Riley. A sherman–morrison–woodbury approach to solving least squares problems with low-rank updates, 2024.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[12] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8026–8037, 2019.

[13] Vincent Plassier, Alexander Fishkov, Victor Dheur, Mohsen Guizani, Souhaib Ben Taieb, Maxim Panov, and Eric Moulines. Rectifying conformity scores for better conditional coverage, 2025.

[14] Yaniv Romano, Evan Patterson, and Emmanuel J. Candès. Conformalized quantile regression, 2019.

[15] Emmanuel J. Candès Aaditya Ramdas Ryan J. Tibshirani, Rina Foygel Barber. Conformal prediction under covariate shift, 2019.

[16] M. Sadinle, J. Lei, and L. Wasserman. Least ambiguous set-valued classifiers with bounded error levels. *Journal of the American Statistical Association*, 114:223–234, 2019.

[17] R. J. Tibshirani, R. Foygel Barber, E. Candes, and A. Ramdas. Conformal prediction under covariate shift. In *Advances in Neural Information Processing Systems 32*, pages 2530–2540, 2019.

[18] V. Vovk. Conditional validity of inductive conformal predictors. In *Proceedings of the Asian Conference on Machine Learning*, volume 25, pages 475–490, 2012.

[19] V. Vovk, I. Nouretdinov, and A. Gammerman. Testing exchangeability on-line. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 768–775, 2003.

[20] Vladimir Vovk, Alex Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. 01 2005.

[21] M. Sesia Y. Romano and E. J. Cand‘es. Classification with valid and adaptive coverage, 2020.

[22] Huan Zhang, Si Si, and Cho-Jui Hsieh. Gpu-acceleration for large-scale tree boosting, 2017.

Appendix

We have tried to implement most of the results and conclusions derived from the paper. Moreover, we have tried to come up with new insights on some of the claims made in the paper. All of our implementation has been using Python, and the codes are available on GitHub.

Some important points to be noted in the implementation:

1. Since marginal coverage is usually computed as an expectation over all possible calibration sets of a fixed size, we have tried to practically simulate this by dividing the entire data set into train data, full_calibration data, and test data.

   Now we try to simulate the action of picking a certain number of elements for a certain number of times (*Calibration set size and sampling iterations are hyperparameters*) and approximate this result to be the expectation.
2. We have used the `MAPIE`[7] library in Python, which allowed us to use a lot of optimized implementations for split conformal algorithms. This definitely helped in reducing the time required to do some of the analysis in split conformal prediction.
3. We have picked models with varying levels of performance and datasets of varying types with multiple classes and difficulty of data points. This helped us get a better view of the robustness of this method of conformal prediction in a wide range of situations.

4.1 Split conformal analysis

In split conformal analysis, we have a trained model, a conformity score function **s**, and a **calibration set**. The calibration set helps in choosing a value of threshold to construct prediction sets as described in the methodology section. The essence of split conformal prediction lies in the guarantee of its marginal coverage, which can be formally stated as follows,

Theorem. (Split conformal coverage guarantee) Suppose $(X_i, Y_i)_{i=1,\dots,n}$ and $(X_{\text{test}}, Y_{\text{test}})$ are exchangeable, and we have a symmetrically constructed score function. Define $\hat{q}$ as the $\lceil (n+1)(1-\alpha) \rceil^{th}$ quantile of the calibration scores $s_1 = s(X_1, Y_1), \cdots, s_n = s(X_n, Y_n)$, and $C(X_{\text{test}}) = \{y : s(X_{\text{test}}, y) \leq \hat{q}\}$. Then the following holds:

$$P\Big(Y_{\text{test}} \in C(X_{\text{test}})\Big) \geq 1 - \alpha.$$

Proof. Since $(X_i, Y_i)_{i=1,\dots,n}$ and $(X_{\text{test}}, Y_{\text{test}})$ are exchangeable, and **s** is symmetric by construction, so is $s_1 = s(X_1, Y_1), ..., s_n = s(X_n, Y_n)$. Thus the rank of $s_{test} = s(X_{test}, y_{test})$ is equally likely to be any number from 1 to $n+1$. We want the probability of this point lying within the $k = \lceil (n+1)(1-\alpha) \rceil^{th}$ quantile, i.e. we want it to be among the k smallest values of $s_1, \cdots, s_n, s_{test}$. This is equal to $\frac{\binom{k}{1} \times n!}{(n+1)!}$ i.e, $k/(n+1)$ or $\lceil (n+1)(1-\alpha) \rceil / (n+1)$. The remainder follows immediately.
For a more general treatment,t refer [15].

To do our experiments, we have used the following two models:

1. **LightGBM**[22]: LightGBM is a gradient-boosting framework that builds models sequentially to minimize prediction errors. It uses decision tree-based learners and focuses on optimizing both accuracy and computational efficiency.
2. **Gaussian Naïve Bayes**: Gaussian Naive Bayes is a probabilistic classification algorithm based on Bayes' theorem with the assumption that features follow a normal (Gaussian) distribution. It treats each feature as independent and models its likelihood using the Gaussian probability density function.

To test the performance of these models, we have used the Forest Covertype data set from UC Irvine Machine Learning Repository[4]. This data set has $5,81,012$ instances, 54 features, and 7 classes.

The reason for choosing these particular models is that we needed one poorly performing model and one model that performs with good accuracy. In this case, the Gaussian Naïve Bayes underperforms. This could be because

- The samples of a particular class need not follow a multivariate normal distribution. Gaussian Naïve Bayes makes this assumption while trying to predict.

- The features might be correlated whereas Gaussian Naïve Bayes assumes they are independent.

To support the above statement, the following are the accuracy results of both models:

- GaussianNB: 48.65%
- LightGBM: 87.19%

These values can be seen by running the code in GitHub

Analysis of Average Prediction set size vs α. From the plot [1]

- The average prediction set size of Gaussian NB is always more than the average prediction set size of the Light GBM model. This is because to satisfy the required coverage ($\mathbb{P} \geq 1 - \alpha$), the poor model includes more classes than required, whereas Light GBM, being more accurate, can attain this marginal coverage by keeping fewer classes in the prediction set.
- For a value of $\alpha < 0.1$ the Gaussian NB model is predicting all the 7 classes in the prediction set which is trivially true to contain y_{test} in it.
- As α increases, the average prediction set size of LightGBM is becoming less than 1. This indicates that for some X_{test}, the prediction set is empty. This is a problem of conformal prediction, i.e., it can provide marginal coverage; however, conditional coverage is difficult to guarantee.

Analysis of Marginal Coverage vs α

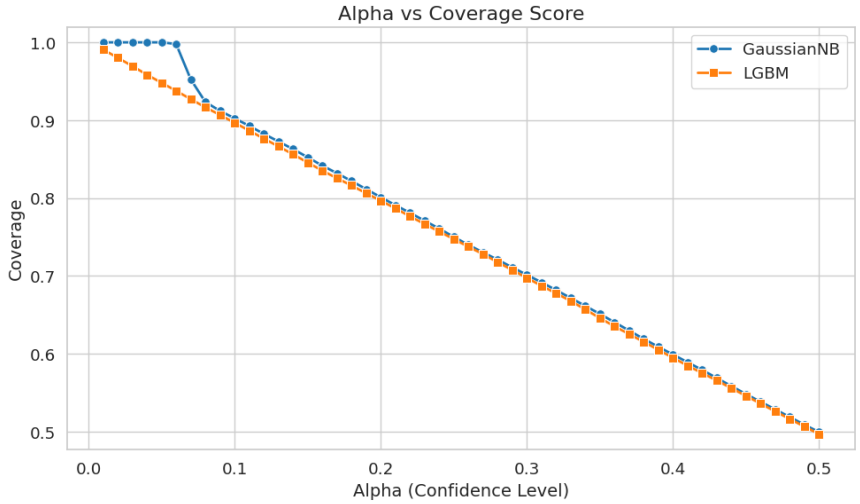

Figure 6: Marginal Coverage vs α of two models

- Irrespective of the choice of model, up to the 2nd decimal, we are getting very similar marginal coverages from both models throughout the range of α. This shows that the guarantee on the marginal coverage is model agnostic.
- For very small values of $\alpha (< 0.1)$, we see that the GaussianNB model is giving a marginal coverage of 100%, and this is because of the fact that it is an underperforming model. This causes it to output the trivial prediction set of all the class labels. However, as α increases, it is able to provide closer fit coverages.

Analysis of Deviation from True Coverage vs α. From the plot [1]

- On taking expectation we see that both models give marginal coverage with accuracy up to 10^{-3}. This again shows that irrespective of the choice of model, marginal coverage is guaranteed by the algorithm.
- We see an increase in coverage in the case of GaussianNB for $\alpha < 0.1$ because it is a poor model, and since it is predicting the entire label set, the coverage is always 100%, and hence it increases.

- LightGBM model is always giving coverage without much significant dependence on α. It appears as if it is giving a negative deviation; however, this is because our approximation of expectation is not good enough. If we carry out the actual condition, we are likely to get very good results.

Analysis of Experiments

In this section, we present an extensive empirical evaluation of split conformal prediction under various modeling and design choices. The goal of these experiments is twofold: to explore the robustness of conformal prediction across different settings, and to gain insights into how different model characteristics and scoring mechanisms influence the resulting prediction sets.

We focus on the well-known CIFAR-10 dataset, which contains 60,000 images evenly distributed across 10 distinct classes. This dataset serves as a benchmark for assessing the conformal prediction framework using multiple classifiers and nonconformity score functions.

For empirical comparison, we primarily use two models of contrasting predictive power:

- **Gaussian Naive Bayes (NB)** — a simple, fast probabilistic model with strong assumptions.

- **LightGBM** — a gradient boosting framework that often provides state-of-the-art performance in tabular and structured data.

Prediction Set Size Distribution vs. α

One key evaluation metric in conformal prediction is the distribution of prediction set sizes for different values of the miscoverage rate α. We compute the size of the conformal prediction set for a random sample of test points and analyze how this distribution evolves as α varies.

This analysis reflects the model's confidence calibration: a strong model should be able to assign smaller prediction sets while still maintaining the required coverage.

Observations:

- For small α values (e.g., $\alpha = 0.01$), the **LightGBM** model shows a desirable distribution in set sizes, indicating that it can achieve tight sets even at high confidence.

- In contrast, the **GaussianNB** model tends to produce uniformly large set sizes (typically of size 6), which is indicative of the model's limited discriminative power. It must include many classes to ensure coverage.

- As α increases, the GaussianNB model starts producing more varied set sizes, indicating improved flexibility in prediction set generation.

- At higher values of α, the LightGBM model often predicts sets of size 1 or 2, leveraging its already high accuracy.

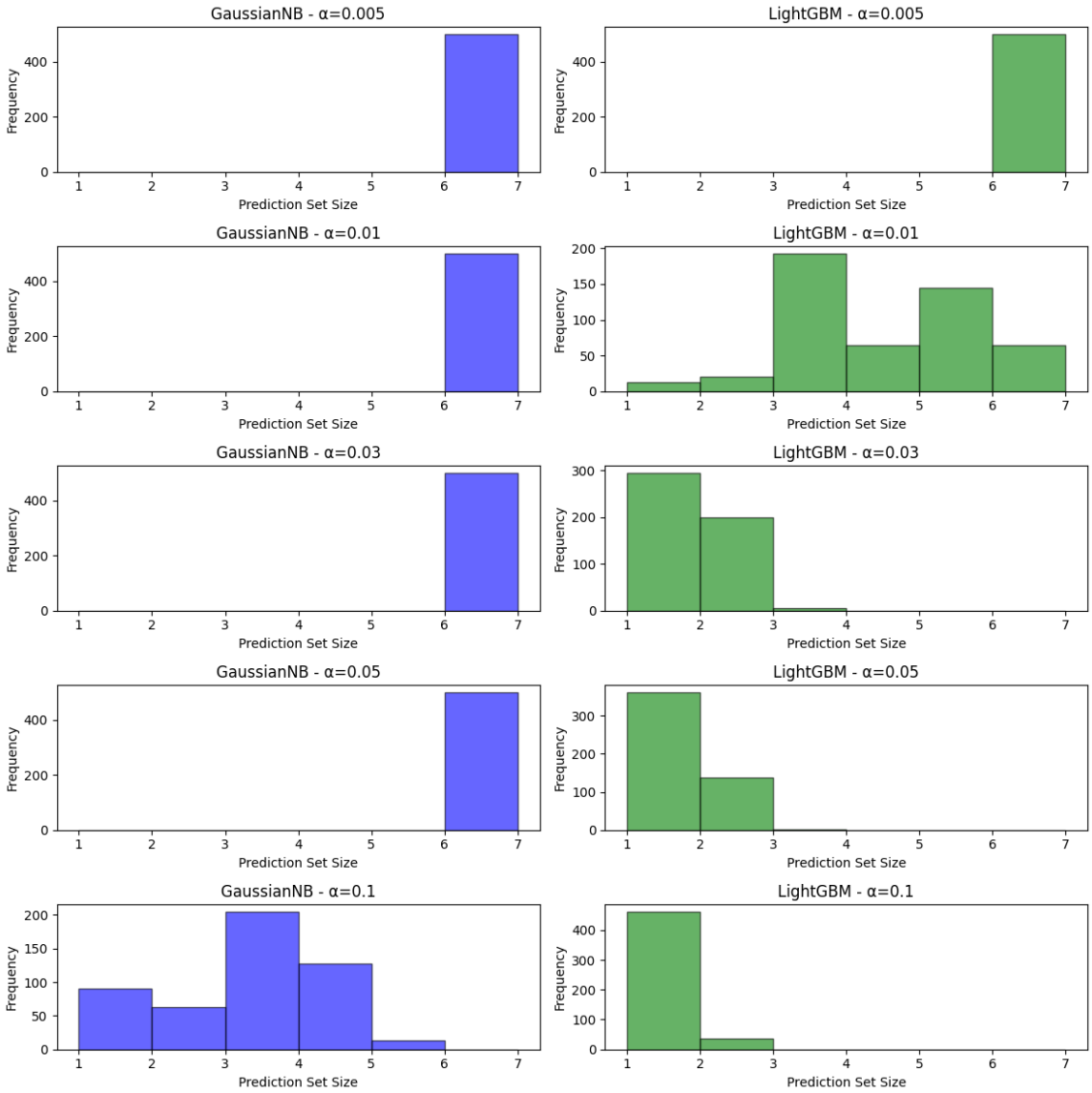

Figure 7: Prediction set sizes for randomly sampled test points across different α values.

This behavior aligns with our expectations — better-performing models like LightGBM can afford to make tighter predictions while still ensuring coverage, whereas weaker models like GaussianNB must compensate by expanding their prediction sets.

Score Function Analysis

Another important dimension in conformal prediction is the choice of nonconformity score function. Different score functions capture uncertainty in different ways and can significantly affect the shape and coverage of the resulting prediction sets.

Refer [3]

Objective: To empirically evaluate how the choice of score function affects the performance of conformal prediction across different models.

Dataset and Setup: We again use the CIFAR-10 dataset, dividing it into training, calibration, and test sets in varying proportions. This also allows us to examine how calibration size affects final prediction quality.

We train multiple classifiers with diverse characteristics, including:

- Decision Tree
- Random Forest
- Logistic Regression
- Naive Bayes

Each model is tested with several commonly used nonconformity score functions, including:

- $s(x, y) = 1 - \hat{P}(y \mid x)$: the basic confidence-based score.
- $s(x, y) = -\log \hat{P}(y \mid x)$: a log-likelihood variant capturing probabilistic sharpness.
- **Entropy**: $s(x) = -\sum_i \hat{P}(i \mid x) \log \hat{P}(i \mid x)$.
- **Top-2 Margin**: $s(x) = \hat{P}(y_1 \mid x) - \hat{P}(y_2 \mid x)$, where y_1 and y_2 are the most and second-most probable classes.

Procedure Outline:

1. Train models of varying capacity and accuracy on the training set.
2. For each model, use different score functions to compute nonconformity scores on the calibration set. Then, compute the empirical quantile $\hat{q}$ corresponding to the desired miscoverage rate α.
3. Construct prediction sets on the test set using:

$$\hat{C}(x) = \{y : s(x, y) \leq \hat{q}\}$$

4. Measure the coverage and average prediction set size for each score function across models.
   Note : This coverage is computed by taking an average value over multiple splits of the datasets and predictions. This is to give us a better notion of the marginal coverage guarantee that is promised as opposed to a single run.

Insights:

- Score functions like Top-2 Margin or Entropy may perform better in low-confidence regions, capturing nuanced uncertainty.
- Simpler scores like $1-\hat{P}(y \mid x)$ may suffice for well-calibrated models but can fail for under-confident or biased models.
- The trade-off between coverage and efficiency (i.e., smaller prediction sets) is directly influenced by the score function's sensitivity to model confidence.

4.1.1 Examples

To illustrate the differences, here are some examples sampled randomly with different models and score functions :


```
===== Model: Random Forest =====

  >> Score Function: 1 - P(y | x)
  Sample 1: True Class = automobile
    Predicted Set = ['airplane', 'automobile', 'bird', 'horse', 'ship', 'truck']

  Sample 2: True Class = deer
    Predicted Set = ['bird', 'cat', 'deer', 'dog', 'frog', 'horse']

  Sample 3: True Class = ship
    Predicted Set = ['airplane', 'automobile', 'ship', 'truck']

  Sample 4: True Class = cat
    Predicted Set = ['automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse']

  Sample 5: True Class = airplane
    Predicted Set = ['airplane', 'automobile', 'bird', 'ship', 'truck']
```

Figure 8: Prediction sets on the CIFAR dataset by the random forest model


```
===== Model: Logistic Regression =====

  >> Score Function: 1 - P(y | x)
  Sample 1: True Class = automobile
    Predicted Set = ['bird', 'horse', 'ship']

  Sample 2: True Class = deer
    Predicted Set = ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck']

  Sample 3: True Class = ship
    Predicted Set = ['airplane', 'automobile', 'deer', 'frog', 'truck']

  Sample 4: True Class = cat
    Predicted Set = ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship']

  Sample 5: True Class = airplane
    Predicted Set = ['airplane', 'automobile', 'bird', 'cat', 'dog', 'horse', 'ship', 'truck']
```

Figure 9: Prediction sets on the CIFAR dataset by the logistic regression model

This setup allows us to compare both the *robustness of conformal prediction under different uncertainty estimates* and the *reliability of score functions* across a spectrum of classifiers.

Effect of Calibration Set Size

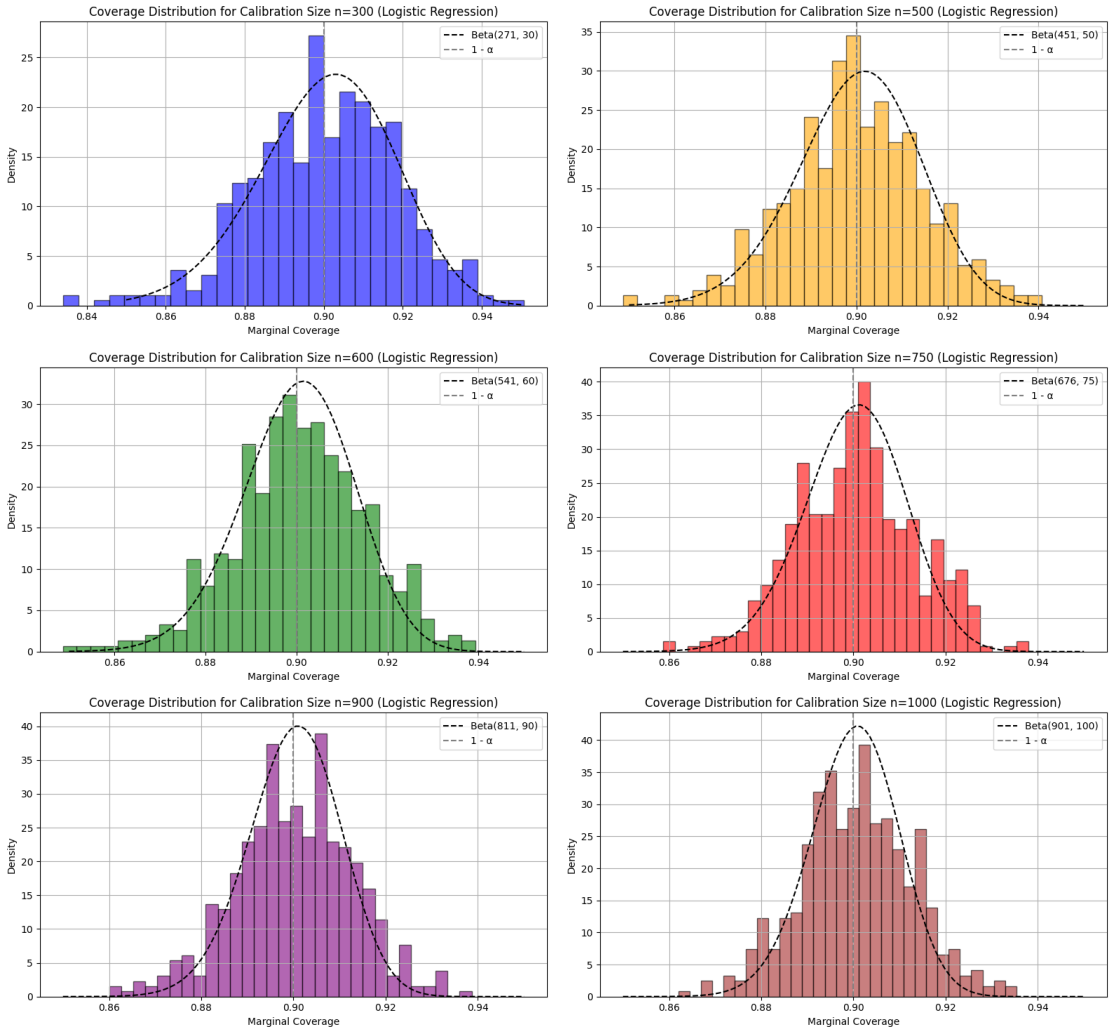

Figure 10: Coverage distribution using various calibration set sizes. From top left : $n_{cal} = 300, 500, 600, 700, 900, 1000$ - Empirical validation of predicted beta distribution

In this experiment, we investigate how the size of the calibration set influences two crucial quantities in conformal prediction:

- The **marginal coverage** — the empirical proportion of test points for which the true label falls within the predicted set.
- The **average prediction set size** — which reflects how informative or efficient the prediction sets are.

Theoretical Background: As shown in [13], the marginal coverage of conformal prediction when using a calibration set of size n follows a **Beta distribution**. More specifically, if α is the target miscoverage level, and the nonconformity scores are exchangeable, then:

$$\mathbb{P}(Y \in \hat{C}(X)) \sim \text{Beta}(n(1-\alpha)+1, n\alpha)$$

This result implies that the marginal coverage is a *random variable*, and its variance is inversely proportional to the size of the calibration set. Hence, larger calibration sets yield more stable and tighter coverage around the desired level $1-\alpha$.

Empirical Evaluation: To validate this, we conduct a series of trials on multiple models and score functions. We vary the size of the calibration set while keeping the training and test splits fixed, and compute:

- The empirical coverage rate across several random test batches.
- The average size of prediction sets over these test points.

For each calibration size n_{cal}, we repeat the experiment multiple times (using different random calibration-test splits) and collect statistics over the resulting coverage values. Refer Figure:[**??**]

Observations:

- For small calibration sets (e.g., $n_{\text{cal}} \leq 100$), the empirical coverage is highly variable, deviating significantly from the nominal level $1 - \alpha$. This is expected due to the high variance in Beta distributions with small sample sizes.
- As n_{cal} increases, the coverage stabilizes and consistently concentrates around the target level. This confirms the theoretical Beta distribution behavior.
- Interestingly, the average prediction set size also tends to decrease slightly with more calibration data, suggesting that the model better estimates nonconformity quantiles and avoids being overly conservative.

Conclusion: These results highlight the importance of having a sufficiently large calibration set for obtaining reliable and tight conformal prediction sets. While conformal prediction offers finite-sample guarantees, the choice of calibration size introduces a practical bias-variance tradeoff between confidence stability and data utilization.

4.2 Conformal prediction on Regression models

Just like how we were able to produce prediction sets in the case of conformal prediction for classification models, we can analogously create prediction ranges in the case of conformal prediction for regression models. To do so, there are several methods. A few of these methods are being discussed here.

Conformalized Quantile Regression

Just like how we accounted for a theoretical guarantee in a classification problem, we can also incorporate a theoretical guarantee while doing regression predictions as well.

Outline of the procedure of prediction:

1. Firstly we aim to create 2 point predictor models $\hat{t}_{\frac{\alpha}{2}}$ and $\hat{t}_{1-\frac{\alpha}{2}}$. The intuition behind this is that let's say we know the underlying distribution of the dataset, and we have the true value of the α quantile and $1 - \alpha$ quantile. Denote this by $t_{\frac{\alpha}{2}}, t_{1-\frac{\alpha}{2}}$.

   Now we know from statistics that

   $$\mathbb{P}(x \in [t_{\frac{\alpha}{2}}, t_{1-\frac{\alpha}{2}}]) = 1 - \alpha$$

   This probabilistic guarantee motivates us to create the point predictors for these quantiles.

2. To create such a point predictor for the quantile, we can construct a normal regression problem with an alternative loss function called the **Quantile Loss/Pinball Loss** which is given by

   $$L_\gamma(\hat{t}_\gamma, y) = (y - \hat{t}_\gamma)\gamma 1_{y > \hat{t}_\gamma} + (\hat{t}_\gamma - y)(1 - \gamma)1_{y \leq \hat{t}_\gamma}$$

   This can be alternatively written as

   $$L_\gamma(\hat{t}_\gamma, y) = max\{(\hat{t}_\gamma - y)(1 - \gamma), \gamma(y - \hat{t}_\gamma)\}$$

   An interesting observation is that on setting $\gamma = 0.5$, we get the standard MSE Loss function

3. Once we have the point predictors, we need a "good" score function so as to get a tight fit on the interval in which the predicted value can lie with a theoretical guarantee. To do so, we define $s(x, y)$ as

   $$s(x, y) = max\{\hat{t}_{\frac{\alpha}{2}} - y, y - \hat{t}_{1-\frac{\alpha}{2}}\}$$

   where $x \in$ Calibration dataset and y corresponds to its value.

4. Now that we have the scores defined for each data point in the calibration data set, we do the same procedure as what we would do in case of split conformal prediction which is to order the scores and set
$$\hat{q} = \frac{\lceil (n+1)(1-\alpha) \rceil}{n} \text{ observation of the set} \{s_1, s_2, s_3, s_4, s_5 ... s_n\}$$
5. Finally, for any x_{test}, we define the interval in which the prediction can lie as
$$C(x) = [\hat{t}_{\frac{\alpha}{2}}(x) - \hat{q}, \hat{t}_{1-\frac{\alpha}{2}}(x) + \hat{q}]$$

This is just one way to arrive at a continuous interval, and there are definitely many other ways; however, there are some advantages as to why Quantile Regression is widely used.

- The interval $[t_{\frac{\alpha}{2}}, t_{1-\frac{\alpha}{2}}]$ by itself gives a good enough coverage by itself.
- This method gives an *asymptotically valid conditional coverage*.

  Conditional Coverage gives us a guarantee for any x_{test} we pick, and on mathematically formalizing it, we get that
  $$\mathbb{P}(y_{test} \in C(x_{test}) | x_{test}) \geq 1 - \alpha$$
  whereas marginal coverage gives us that on an average among all x_{test}, we have that
  $$\mathbb{P}(y_{test} \in C(x_{test})) \geq 1 - \alpha$$
- The quantile loss function can be very easily modified to be incorporated in any regression model. In fact, the the standard MSE is just a specific case of the quantile loss function, so we can easily interconvert between MSE Loss and Quantile Loss function

Conformalizing Scalar Uncertainty Estimates

This method rather than trying to create a confidence interval like quantile regression is trying to create an uncertainty interval around the point predictor so that the coverage guarantee still holds.

So in this type, we have a point predictor $\hat{f}(x)$ and an uncertainty predictor $\hat{u}(x)$. This function $\hat{u}(x)$ could represent standard deviation, variance, residuals or any type of function that is *negatively oriented*. This $\hat{u}(x)$ should be a good estimate of the function $u(x)$ it is trying to estimate so in reality, we would do some regression model on this parameter as well.

Now that we have these two predictor functions, we can define the score function for the calibration data as :
$$s(x, y) = \frac{|y - \hat{f}(x)|}{\hat{u}(x)}$$
Once we have this score function, we do the same procedure as sorting them in ascending order, getting the $\frac{\lceil (n+1)(1-\alpha) \rceil}{n}$ quantile and setting it to $\hat{q}$.

Once we have found out $\hat{q}$, for any x_{test}, we can define the prediction interval to be
$$C(x) = [\hat{f}(x) - \hat{u}(x)\hat{q}, \hat{f}(x) + \hat{u}(x)\hat{q}]$$
This prediction interval gives us the same coverage guarantee. When the function $\hat{u}(x)$ represents the predictor of uncertainty/ spread at the point x, we call the method **Studentized Residual Method**

Proof of the marginal coverage of both the above methods can be found in [14], [22]

Remark:
It is preferred to use quantile regression over uncertainty estimates because in quantile regression the predictors $\hat{t}_{\frac{\alpha}{2}}$ is directly related to the significance level α whereas in uncertainty estimates, both the predictors $\hat{f}(x), \hat{u}(x)$ are not related to α and hence we are not giving more importance to this parameter as much as we do in quantile regression.

In practice, quantile regression seems to perform slightly better than uncertainty estimation methods and this is has been cited in one of Angelopoulos' paper.

Nevertheless, uncertainty estimates are very easy to implement and very intuitive to understand, so it is still widely used.

Full Conformal Prediction

This method is different from the above methods in the sense that there is no calibration data set. We only have the usual train and test data set; however, the catch is that it is computationally more expensive. The tradeoff is between computational expense(*In some special cases, this is not the case*) and more train data.

Full conformal prediction is a technique used to construct prediction intervals with valid coverage guarantees, assuming the data are exchangeable. Unlike split conformal prediction (which uses a fixed calibration set), full conformal prediction includes the test point (x, y) as part of the model training process.

To determine whether a candidate value $y \in \mathbb{R}$ belongs to the prediction set for a new input x, we augment the dataset with (x, y), train a model on all $n + 1$ points, and compute the residuals:

$$R_i^{(x,y)} = |Y_i - \hat{f}_{(x,y)}(X_i)|, \quad i = 1, \ldots, n,$$
$$R_{n+1}^{(x,y)} = |y - \hat{f}_{(x,y)}(x)|.$$

Then, y is included in the prediction set $\hat{C}_n(x)$ if its residual $R_{n+1}^{(x,y)}$ is among the smallest $\lceil (1-\alpha)(n+1) \rceil$ of the residuals $\{R_1^{(x,y)}, \ldots, R_{n+1}^{(x,y)}\}$.

This method guarantees that the coverage probability satisfies:

$$\mathbb{P}\left(Y_{n+1} \in \hat{C}_n(X_{n+1})\right) \in \left[1-\alpha, 1-\alpha+\frac{1}{n+1}\right],$$

under the assumption that the residuals are almost surely distinct. The key idea is that, because the residuals are exchangeable under the inclusion of the test point, we can use rank-based reasoning to ensure valid coverage. In place of the absolute residual score, any negatively oriented function can be used along with a suitable symmetric score function, and the guarantee of the marginal coverage will still hold.

Proof of marginal coverage of full conformal prediction can be found in [20]

Jackknife, Jackknife+

These methods are very closely related to full conformal prediction in the sense that it doesn't involve a calibration data set that is derived from the train data set, and also, they involve the process of retraining a large number of times, which makes it extremely computationally expensive.

Jackknife Prediction Intervals

We begin by introducing quantile notation. For any values $v_1, \ldots, v_n$, define the upper and lower empirical quantiles as:

$$\hat{q}^+_{n,\alpha}\{v_i\} = \text{the } \lceil (1-\alpha)(n+1) \rceil \text{ smallest value of } \{v_1, \ldots, v_n\},$$
$$\hat{q}^-_{n,\alpha}\{v_i\} = \text{the } \lfloor \alpha(n+1) \rfloor \text{ smallest value of } \{v_1, \ldots, v_n\}.$$

Note that $\hat{q}^-_{n,\alpha}\{v_i\} = -\hat{q}^+_{n,\alpha}\{-v_i\}$, which gives a simple relationship between the upper and lower quantiles.

Let $\hat{\mu}$ be a regression function trained using a supervised learning algorithm A on the dataset $(X_1, Y_1), \ldots, (X_n, Y_n)$. Formally, we define:

$$\hat{\mu} = A((X_1, Y_1), \ldots, (X_n, Y_n)).$$

Using this, the naive conformal prediction interval is given by:

$$\hat{C}^{\text{naive}}_{n,\alpha}(X_{n+1}) = \hat{\mu}(X_{n+1}) \pm \hat{q}^+_{n,\alpha}\left\{|Y_i - \hat{\mu}(X_i)|\right\}.$$

For the jackknife method, we define the leave-one-out model $\hat{\mu}_{-i}$ as the model trained on all points except the i-th:

$$\hat{\mu}_{-i} = A((X_1, Y_1), \ldots, (X_{i-1}, Y_{i-1}), (X_{i+1}, Y_{i+1}), \ldots, (X_n, Y_n)).$$

The corresponding leave-one-out residual is:

$$R_i^{\text{LOO}} = |Y_i - \hat{\mu}_{-i}(X_i)|.$$

Then, the jackknife conformal prediction interval is:

$$\hat{C}_{n,\alpha}^{\text{jackknife}}(X_{n+1}) = \hat{\mu}(X_{n+1}) \pm \hat{q}_{n,\alpha}^{+}\left\{R_i^{\text{LOO}}\right\}.$$

Lastly, the learning algorithm A is assumed to be symmetric with respect to the data: its output should remain unchanged under any reordering of the training examples. Formally, for any permutation π of $\{1, \ldots, m\}$, we require:

$$A((X_{\pi(1)}, Y_{\pi(1)}), \ldots, (X_{\pi(m)}, Y_{\pi(m)})) = A((X_1, Y_1), \ldots, (X_m, Y_m)).$$

This assumption will be made in the case of Jackknife+ as well

Jackknife+ Prediction Intervals

The jackknife+ method improves upon the classical jackknife conformal interval by using leave-one-out predictions not only for computing residuals but also for predicting the test point. As before, let $\hat{\mu}_{-i}$ denote the prediction function trained on the dataset with the i-th point removed.

The jackknife+ prediction interval for a new input X_{n+1} is defined as:

$$\hat{C}_{n,\alpha}^{\text{jackknife+}}(X_{n+1}) = \left[\hat{q}_{n,\alpha}^{-}\left\{\hat{\mu}_{-i}(X_{n+1}) - R_i^{\text{LOO}}\right\}, \hat{q}_{n,\alpha}^{+}\left\{\hat{\mu}_{-i}(X_{n+1}) + R_i^{\text{LOO}}\right\}\right],$$

where $R_i^{\text{LOO}} = |Y_i - \hat{\mu}_{-i}(X_i)|$ is the leave-one-out residual for the i-th data point, and $\hat{q}_{n,\alpha}^{\pm}$ are the empirical lower and upper quantiles defined as:

$$\hat{q}_{n,\alpha}^{+}\{v_i\} = \text{the } \lceil (1-\alpha)(n+1) \rceil \text{ smallest value of } \{v_1, \ldots, v_n\},$$

$$\hat{q}_{n,\alpha}^{-}\{v_i\} = \text{the } \lfloor \alpha(n+1) \rfloor \text{ smallest value of } \{v_1, \ldots, v_n\}.$$

For comparison, the classical jackknife interval centers the prediction interval around $\hat{\mu}(X_{n+1})$, the model trained on the full dataset:

$$\hat{C}_{n,\alpha}^{\text{jackknife}}(X_{n+1}) = \left[\hat{q}_{n,\alpha}^{-}\left\{\hat{\mu}(X_{n+1}) - R_i^{\text{LOO}}\right\}, \hat{q}_{n,\alpha}^{+}\left\{\hat{\mu}(X_{n+1}) + R_i^{\text{LOO}}\right\}\right].$$

While both methods use the same leave-one-out residuals R_i^{LOO}, the jackknife+ interval is more adaptive by using $\hat{\mu}_{-i}(X_{n+1})$, the prediction from the model that did not see the i-th data point, thus maintaining a better symmetry between training and test.

While the jackknife procedure is widely used for constructing prediction intervals, it lacks universal theoretical guarantees. Most known results rely on asymptotic analyses or assume a certain degree of stability in the underlying regression estimator $\hat{\mu}$. In settings where $\hat{\mu}$ is unstable—such as when the sample size n is close to the data dimension d—the jackknife method can fail to provide valid coverage. For example, our simulations (see Section 7) demonstrate that jackknife intervals may exhibit extremely poor coverage when used with ordinary least squares in high-dimensional regimes.

To address these shortcomings, the jackknife+ method has been proposed as a refinement of the classical jackknife. Unlike its predecessor, the jackknife+ provides non-asymptotic coverage guarantees without requiring any assumptions beyond the exchangeability of the training and test data. In the worst-case scenario, it guarantees a coverage level of at least $1 - 2\alpha$, whereas the jackknife method may yield arbitrarily poor or even zero coverage in degenerate cases. Nevertheless, in many practical situations, especially when the regression algorithm is stable, the jackknife and jackknife+ methods tend to produce similar intervals, and both approach the desired coverage level of $1 - \alpha$. From a theoretical standpoint, under suitable stability conditions, both methods can be shown to deliver nearly the same level of predictive coverage.

Remark: In the case of Jackknife and Jackknife+, when we are doing the usual linear regression, instead of always retraining the model, we can use the *Sherman Morrison Update*[10] to efficiently remove a point from training and retraining. This helps in reducing the time complexity by a big margin. This particular implementation has been done, and the code for it is present in the GitHub page.

Proof of coverage guarantees can be found in [2].

Implementations of conformal prediction

To do this task, we have used Neural Network models and XGBoost as our main models. The dataset we have used in this case is the California Housing Data set which has $20,640$ samples and 8 features.

- We have implemented the standard quantile regression model using neural networks
- We have implemented the studentized conformal model using neural networks; however, the calculation is very slow.
- The same process of computing the studentized conformal model is made faster by using the XGBoost model. This is because we can use GPU acceleration as well as compute uncertainty easily by only computing variance in the leaves of the tree.
- We have implemented the Jackknife/Jackknife+ as mentioned in [2]. Code is available in the GitHub repository.

Analysis of deviation from true marginal coverage of models [1]

1. We see that the deviation is increasing(positive deviation) in case of quantile regression with increase in α whereas studentized model is almost invariant. This affirms the claim made earlier that since the quantile regression is more dependant on α parameter, we see larger changes in quantile regression.
2. The fact that quantile regression's marginal coverage is always more than the studentized model affirms the fact that the quantile regression model is performing better in giving better marginal coverage because of its direct dependence on the α parameter.

4.3 Extensions to conformal prediction

We try extending the notion of conformal prediction to more general and practical scenarios and see what additional constraints or problems would have to deal with in it's implementation.

Group-Balanced Conformal Prediction

As an extension to conditional coverage for this method that we highly desire, in the more general setting, we require certain groups in specific to satisfy the coverage guarantee at least. E.g. medical reports predicted from different ethnicities or groups of people.

Formally, Let the inputs $X_{i,1},\ i = 1, \ldots, n$, take values in a discrete set $\{1, \ldots, G\}$, corresponding to categorical groups. We then require group-balanced coverage:

$$\mathbb{P}\left(Y_{\text{test}} \in \mathcal{C}(X_{\text{test}}) \,|\, X_{\text{test},1} = g\right) \geq 1 - \alpha,$$

for all groups $g \in \{1, \ldots, G\}$.

Note: These groups may be partitioned after the calibration. Since we cannot guarantee the conditional coverage in each of the groups, we make a slight change to our approach: We run the conformal prediction over each of the groups separately.

We do this by computing the conformal quantile for each of the groups within themselves and then running the conformal algorithm. Let $n(g)$ denote the number of calibration examples belonging to the group g. We define the group-specific quantile $q(g)$ as:

$$q(g) = \text{Quantile}\left(s_1, \ldots, s_{n(g)}; \lceil (n(g)+1)(1-\alpha) \rceil / n(g)\right),$$

where s_i are the conformity scores of the calibration points in group g.

Finally, we form the prediction set $\mathcal{C}(x)$ by selecting the appropriate quantile based on the group membership of x:

$$\mathcal{C}(x) = \{y \in \mathcal{Y} : s(x, y) \leq \hat{q}(x_1)\},$$

where x_1 denotes the group attribute of x, and $\hat{q}(x_1)$ is the threshold corresponding to that group.

This $\mathcal{C}$ satisfies the required coverage property across the various groups that we desire.

Class-Conditional Conformal Prediction

Since conditional coverage is not achievable, we can try to achieve marginal coverage over every class as a subset. This process of achieving class-conditional coverage requires slight modifications compared to regular conformal prediction methods.

As seen in evaluation methods, we can split the calibration data set into groups where each group corresponds to a particular class.

For a given class k, we can define the score of the ith occurrence in this class as

$$s_i^k = s(X_j, y_j)$$

This means that after splitting into groups, the jth data point is the ith data point in the kth class.

Now for each class, we can define the quantile to be

$$\hat{q}^k = \frac{\lceil (n^k + 1)(1 - \alpha) \rceil}{n^k} \text{ Quantile of the set} \{s_1^k, s_2^k, ...s_{n^k}^k\}$$

where n^k is the number of data points in the kth class. Now we can define the prediction set to be

$$C(x) = \{y : s(x, y) \leq \hat{q}^y\}$$

The proof of why this choice of prediction set satisfies marginal coverage can be found in [16, 18]

Conformal Risk control

For most machine learning problems, we don't necessarily bound our generalization loss just by the expression :

$$E[1_{Y_{test} \notin C(X_{test})}]$$

But we may use a general loss function on this value. Thus we would want a guarantee of the form :

$$\mathbb{E}\left[\mathbb{I}\left\{Y_{\text{test}} \notin C(X_{\text{test}})\right\}\right] \leq \alpha$$

We show that conformal prediction, in general, can even provide this guarantee for some general and arbitrary bounded loss function *l* if we tweak our algorithm a bit.

In practice, we often apply a post-processing step to a base model f in order to construct a prediction set $C_\lambda(\cdot)$. The parameter λ influences the expressiveness of the prediction model - typically, a larger value of λ gives a bigger prediction set.

To evaluate the performance of such prediction sets, we consider a loss function $\ell(C_\lambda(x), y)$, which is bounded above by a constant $B < \infty$ and is non-increasing in λ. That is, increasing λ (making the prediction set more conservative) does not worsen the loss.

We define the empirical risk over a calibration set of size n as:

$$\hat{R}(\lambda) = \frac{1}{n} \sum_{i=1}^{n} \ell(C_\lambda(X_i), Y_i).$$

To ensure valid risk control, we select the smallest value of λ, denoted $\hat{\lambda}$, such that the empirical risk is at most a slightly adjusted threshold:

$$\hat{\lambda} = \inf \left\{ \lambda : \hat{R}(\lambda) \leq \alpha - \frac{B - \alpha}{n} \right\}.$$

This adjustment makes the procedure slightly more conservative than simply using the threshold α, which accounts for finite-sample effects. For example, when $B = 1$, $\alpha = 0.1$, and $n = 1000$, the threshold becomes 0.0991 instead of 0.1, thus promoting a more robust selection of $\hat{\lambda}$.

Outlier Detection

There are several distance-based methods, clustering-based methods, density-based methods, etc., to detect outliers in unsupervised learning. Outliers are points that do not belong to the same dataset and enter the dataset due to some error in measurement or some other reason.

These methods associated with detecting outliers have some uncertainty involved, and we can use conformal prediction to quantify this uncertainty.

Using conformal prediction, the main goal is to reduce the False positives because we wouldn't want to loose training data by considering it to be an outlier.

The procedure is not different at all from the standard methods:

1. Assume we have a model for outlier detection with the criteria that a **Large Score from model** $\implies$ **Outlier**. This basically means that we have a negatively oriented score function.
2. Use this score function $s_i = s(X_i)$ on the calibration data set to get the sorted score values.
3. Now define $\hat{q}$ to be the $\frac{\lceil (n+1)(1-\alpha) \rceil}{n}$
4. For every test point, we define the prediction (*Note that here we are making predictions and not prediction set*)
$$C(x) = sign(s(x) - \hat{q})$$
Where $C(x) \leq 0 \implies$ inlier and $C(x) > 0 \implies$ outlier.

Proof of marginal coverage

Refer [19, 3, 8]

Conformal Prediction Under Covariate Shift

In all the methods, the score values are computed on the calibration data and they are simply extended to the test set with the assumption that the probability distribution of the calibration data set is the same as the probability distribution of the test data set.

However, this might not always be true. **Covariate Shift** is a special case where the distribution of (X_{test}, y_{test}) is different from (X_i, y_i) however the distribution of $X_{test}|y_{test}$ is **EQUAL** to the distribution of $X_i|y_i$.

To get better results, we can use weighted conformal prediction as in [17]. The basic intuition is that we now have weights corresponding to each data point, and we would increase the weights of those points that are more likely to be in the test set distribution and reduce the weights of those that are less likely.

A simple choice of weight function for this would be related to the likelihood ratio:
$$w(x) = \frac{f_{test}(x)}{f(x)}$$
Where $f_{test}(x)$ is the probability density of the test data set, and $f(x)$ is the probability density function of the train data set. Now we can define the updated weights as:
$$p_i^w(x) = \frac{w(X_i)}{\sum_{j=1}^{n} w(X_j) + w(x)}$$
Now we do the usual adaptive conformal prediction based on cumulative sums of scores where the scores are going to be re-calculated as follows:
$$\hat{q}(x) = min(s_j : \sum_{i=1}^{j} p_i^w(x) s_i(x) \geq 1 - \alpha)$$
Note that $\hat{q}(x)$ is now dependent on the test data point we pick.

Finally, define the prediction set as
$$C(x) = \{y : s(x, y) \leq \hat{q}(x)\}$$

Proof of marginal coverage in [17]

4.4 RAPS

Presented in [1] we implement the Regularized Adaptive Prediction Set(**RAPS**) algorithm which guarantees to attain a lesser set size while maintaining a smaller set size than a `Naive`(given below) or APS (Adaptive Prediction Set) approach. We'll build the intuition for using this particular method below.

To achieve the marginal coverage guarantee as stated in Section [2], a naive approach(term it `Naive` method) would be -

- Sort the classes as per the softmax scores.
- Greedily include classes starting from the most likely class until the cumulative sum exceeds $1 - \alpha$.

However, the above does not take into account the poor calibration of modern machine learning models ([6]). Thus there is no distribution and model agnostic guarantee that can be given. We have seen that this particular problem is addressed by split conformal prediction, by using a separate holdout/calibration set that adjusts the threshold based on a conformal score function. Even though the score function is largely a design choice, the only constraint being its symmetry, some score functions perform better than others.
The APS procedure described in [21] uses the cumulative sum of sorted softmax scores up until the true label for instance, and evidence provided suggests that this gives superior "adaptivity", i.e. larger prediction sets for "harder" and smaller prediction sets for "easier" instances. The proof of marginal coverage is also a subject of the above. For the algorithm, refer to Section 2.2 in [21].

However, APS faces a challenge in practice- the average set size is quite large. Deep learning classifiers suffer from a permutation problem: the scores of the **less confident classes** (e.g. classes 10 through 1000 in Imagenet-V2) are not reliable probability estimates. The ordering of these classes is primarily determined by 'noise', so APS has to take very large sets for some difficult images.
To address this problem, the **Regularized Adaptive Prediction Sets (RAPS)** method was introduced. We shall devote the remainder of this section building up to RAPS.

Nested Conformal Prediction

Firstly we present a slightly modified version of conformal prediction that is equivalent. Unlike the split conformal prediction we have familiarized ourselves with, nested conformal prediction starts with a sequence of all possible prediction sets $\{\mathcal{F}_t(x)\}_{t\in\mathcal{T}}$ for some ordered set $\mathcal{T}$. The sequence $\mathcal{F}$ is basically a design choice, just as the conformity scores were in CP. These prediction set sequences have the following properties;

- They are "nested" in the sense that for every $t_1 \leq t_2 \in \mathcal{T}$ we have $\mathcal{F}_{t_1} \subseteq \mathcal{F}_{t_2}$.
- $\mathcal{F}_{inf\,\mathcal{T}} = \phi$ and $\mathcal{F}_{sup\,\mathcal{T}} = \mathcal{Y}$ where $\mathcal{Y}$ is the output class of the black-box model.

Then we try to find the smallest $t \in \mathcal{T}$ s.t.

$$\mathbb{P}(Y \in \mathcal{F}_t(X)) \geq 1 - \alpha$$

where α is the user-defined confidence level. This is achieved by using standard split conformal prediction as follows;

- Define conformal score function $r(x, y) := \inf\{t \in \mathcal{T} : y \in \mathcal{F}_t(x)\}$.
- Define the scores for the calibration data $\{r_i = r(X_i, Y_i)\}_{i\in D_2}$ and set

  $$Q_{1-\alpha}(r, D_2) := \lceil (1-\alpha)(1 + 1/|D_2|)\rceil\text{-th quantile of } \{r_i\}_{i\in\ D_2}.$$

  (that is, $Q_{1-\alpha}(r, D_2)$ is the $\lceil (1-\alpha)(1 + 1/|D_2|)\rceil$-th largest element of the set $\{r_i\}_{i\in D_2}$).
- Get the prediction sets as $C(x) := \mathcal{F}_{Q_{1-\alpha}(r,D_2)}(x) = \{y \in \mathcal{Y} : r(x, y) \leq Q_{1-\alpha}(r, D_2)\}$.

Refer [5] for marginal coverage guarantee of nested conformal prediction and equivalence of nested conformal prediction with standard split conformal prediction.

Conformal Calibration - A generalized setting of RAPS

Consider **ANY** procedure which outputs a prediction set given an input instance and is endowed with a $\mathcal{T}$ (call it a 'tuning parameter') that regulates the size of the sets (In APS $\mathcal{T}$ was the cumulative sum of sorted softmax scores).
We now try to choose $\mathcal{T}$ with the help of a calibration set s.t. The prediction sets give marginal coverage. This process is described formally below;

Let $(X_i, Y_i)_{i=1,\ldots,n}$ be an i.i.d. calibration set. Further, let $C(x, u, \mathcal{T}) : R^d \times [0,1] \times R \to 2^{\mathcal{Y}}$ be a **'set-predictor'** function that takes a feature vector x to a subset of the possible labels. The construction of C for each given feature vector is basically what is outlined by **the** procedure. The second argument u is included to allow for randomized procedures, which is necessary to achieve exact coverage (explained further in RAPS section). Suppose that the sets are indexed by $\mathcal{T}$ such that they are **nested**, meaning larger values of $\mathcal{T}$ lead to larger sets:

$$C(x, u, \mathcal{T}_1) \subseteq C(x, u, \mathcal{T}_2) \quad \text{if} \quad \mathcal{T}_1 \leq \mathcal{T}_2 \tag{1}$$

Our goal is to find a value of $\mathcal{T}$ that will achieve $1-\alpha$ coverage on test data. Consider the following candidate;

$$\hat{\mathcal{T}}_{\text{ccal}} = \inf\left\{\mathcal{T} : \frac{|\{i : Y_i \in C(X_i, U_i, \mathcal{T})\}|}{n} \geq \frac{\lceil (n+1)(1-\alpha) \rceil}{n}\right\} \tag{2}$$

The set function $C(x, u, \mathcal{T})$ with this $\mathcal{T}_{ccal}$ is guaranteed to have finite-sample coverage on a fresh test sampling, as stated formally next.

Theorem Suppose $(X_i, Y_i, U_i)_{i=1,\ldots,n}$ and $(X_{n+1}, Y_{n+1}, U_{n+1})$ are i.i.d. and let $C(x, u, \mathcal{T})$ be a 'set-predictor' function satisfying the nesting property. Suppose further that the sets $C(x, u, \mathcal{T})$ grow to include all labels for large enough $\mathcal{T}$ i.e. for all $x \in R^d$, $C(x, u, \mathcal{T}) = \mathcal{Y}$ for some $\mathcal{T}$. Then for $\hat{\mathcal{T}}_{ccal}$ defined above, we have the following,

$$P\Big(Y_{n+1} \in C(X_{n+1}, U_{n+1}, \hat{\mathcal{T}}_{ccal})\Big) \geq 1 - \alpha.$$

Proof. Refer [1].

RAPS

We noted that conformal calibration was for ANY procedure that gave a prediction set with some input feature vector and had a tuning parameter. We shall now consider a special case;

Formally, let $\rho_x(y) = \sum_{y'=1}^{K} \hat{\pi}_x(y') \mathbb{I}_{\{\hat{\pi}_x(y') > \hat{\pi}_x(y)\}}$ be the total probability mass of the set of labels that are more likely than y. These are all the labels that will be included before y is included. In addition, let $o_x(y) = |\{y' \in \mathcal{Y} : \hat{\pi}_x(y') \geq \hat{\pi}_x(y)\}|$ be the ranking of y among the label based on the probabilities $\hat{\pi}$. We take

$$C^*(x, u, \mathcal{T}) := \Big\{y \ : \ \rho_x(y) + \hat{\pi}_x(y) \cdot u + \underbrace{\lambda \cdot |(o_x(y) - k_{reg})|}_{\text{regularization term}} \leq \mathcal{T}\Big\} \tag{3}$$

where $\lambda,\ k_{reg} \geq 0$ are regularization hyperparameters that are helpful in preventing large prediction sets.

Remarks:

1. $\rho_x(y)$ increases as y ranges from the most probable to least probable label, so our sets will prefer to include the y that are predicted to be the most probable by black-box model.
2. $\hat{\pi}_x(y) \cdot u$ is a randomized term to handle the fact that the value will jump discretely with the inclusion of each new y. The randomization term can never impact more than one value of y since there is at most one value of y such that $y \in C(x, 0, \mathcal{T})$ but $y \notin C(x, 1, \tau)$. The following example illustrates the need for randomized predictors,
   ex. Assume for a particular input image we expect a set of size k to have 91% coverage, and a set of size $k-1$ to have 89% coverage. In order to achieve our desired coverage of 90%, we randomly choose size k or $k-1$ with equal probability. In general, the probabilities will not be equal but rather chosen so the weighted average of the two coverages is exactly 90%.

3. The regularization promotes small set sizes i.e. for values of y that occur farther down the ordered list of classes, the term $\lambda \cdot (o_x(y) - k_{reg})^+$ makes that value of y require a higher value of τ before it is included in the predictive set. For example, if $k_{reg} = 50$, then the 6^{th} most likely value of y has an extra penalty of size 44λ, so it will never be included until $\mathcal{T}$ exceeds $(\rho_x(y) + \hat{\pi}_x(y) \cdot u + 44\lambda)$, whereas it enters when $\mathcal{T}$ exceeds $(\rho_x(y) + \hat{\pi}_x(y) \cdot u)$ in the non-regularized version. Intuitively, a high λ value discourages sets large than k_{reg}

Following formally states RAPS coverage guarantee,

Theorem Suppose $(X_i, Y_i, U_i)_{i=1,...,n}$ and $(X_{n+1}, Y_{n+1}, U_{n+1})$ are i.i.d. and let $C^*(x, u, \mathcal{T})$ be defined above. Suppose further that $\hat{\pi}_x(y) > 0$ for all x and y. Then for $\hat{\mathcal{T}}_{ccal}$ defined as in section 4, we have the following coverage guarantee,

$$1 - \alpha \leq P\Big(Y_{n+1} \in C^*(X_{n+1}, U_{n+1}, \hat{\mathcal{T}}_{ccal})\Big) \leq 1 - \alpha + \frac{1}{n+1}.$$

This is a corollary of the theorem establishing marginal coverage of Conformal Calibration methods.

Implementation

In our implementation, we took the pre-trained model **ResNet152**[11] from the `torchvison` library as in [12] with standard normalization, resize, and crop parameters. Before applying `Naive`, **APS**, or **RAPS**, we calibrated the classifiers using the standard Temperature scaling(Platt scaling for Binary classes) as given in [9]. The Procedure is split into Conformal Calibration for **RAPS**, as mentioned in Appendix [4.4] to get the generalized quantile $\hat{\tau}_{ccal}$ via Algorithm [2]. This generalized quantile is then used to get the Prediction Set with (1 - α) confidence, $\mathcal{C}$, via Algorithm [3].

Algorithm 2 RAPS Conformal Calibration

Require: α; $s \in [0,1]^{n \times K}$, $I \in \{1, ..., K\}^{n \times K}$, and $y \in \{0, 1, ..., K\}^n$ corresponding respectively to the sorted scores, the associated permutation of indexes, and ground-truth labels for each of n examples in the calibration set; k_{reg}; λ; boolean $rand$

RAPSC(α, s, I, y, λ):
for $i = 1$ to n **do**
  $L_i \leftarrow j$ such that $I_{i,j} = y_i$
  $E_i \leftarrow \sum_{j=1}^{L_i} s_{i,j} + \lambda(L_i - k_{reg})^+$
  if $rand$ **then**
    $U \sim \text{Unif}(0, 1)$
    $E_i \leftarrow E_i - U \cdot s_{i,L_i}$
$\hat{\tau}_{ccal} \leftarrow$ the $\lceil (1-\alpha)(1+n) \rceil$ largest value in $\{E_i\}_{i=1}^n$
return $\hat{\tau}_{ccal}$

Ensure: The generalized quantile, $\hat{\tau}_{ccal}$ {The value in [2]}

Note: Adapted from [1] Section 2

Algorithm 3 RAPS Prediction Sets

Require: α, sorted scores s and the associated permutation of classes I for a test-time example, $\hat{\tau}_{ccal}$ from Algorithm [2], k_{reg}, λ, boolean $rand$

RAPS($\alpha, s, I, \hat{\tau}_{ccal}, k_{reg}, \lambda, rand$):
$L \leftarrow |\ \{j \in \mathcal{Y} \ :\ \sum_{i=1}^{j} s_i + \lambda(j - k_{reg})^+ \leq \hat{\tau}_{ccal}\}\ | + 1$
if $rand$ **then**
  $U \leftarrow \text{Unif}(0, 1)$
  $L \leftarrow L - \mathbb{I}\Big\{(\sum_{i=1}^{L} s_i + \lambda(L - k_{reg})^+ - \hat{\tau}_{ccal})/(s_L + \lambda \mathbb{I}(L > k_{reg})) \leq U\Big\}$
return $\mathcal{C} = \{I_1, \ldots, I_L\}$ {The L most likely classes}

Ensure: The $1 - \alpha$ confidence set, $\mathcal{C}$ {The set in [3]}

Note: Adapted from [1] Section 2

We have *rand* variable to ensure tie-breaking among the score functions. We keep an additional variable in the Code, called `allow_zero_sets`, which basically allows zero sets to be there in the prediction set.

We compare our RAPS procedure in Section [3.4.1] with the `Naive` procedure and **APS**(Check its just RAPS with $\lambda = 0$) procedure with the model ResNet152 with 3 different values of $\alpha = \{0.1, 0.05, 0.01\}$ on the Dataset ImageNet-Val(by UCB). Clearly, see that **APS** creates a larger data set than the other procedure (While we got lucky for $\alpha = \{0.05, 0.1\}$ to have a set size as equal to Naive). But our RAPS procedure not only gave a lower average set size by a huge margin but also gave the desired coverage.

Hyperparameter selection for RAPS

This requires an extra data splitting step, where a small amount of 'tuning data' $\{x_i, y_i\}_{i=1}^{m}$ is used to estimate k^*, and then k_{reg} is set to k^*. The algorithm [4] is used for selecting k_{reg}. For λ, a simple grid search followed by a selection of the value such that achieves the smallest size of prediction sets for given k^* on the holdout (tuning) set of size m suffices.

Method | λ	0.001	0.01	0.1	0.5	1.0
RAPS Avg Size	11.539	5.863	4.256	6.034	6.033
RAPS Coverage	0.965	0.963	0.955	0.956	0.956

Table 3: Grid search on values of λ

Algorithm 4 Adaptive Fixed-K

Require: α; $I \in \{1, ..., K\}^{n \times K}$, and $y \in \{0, 1, ..., K\}^n$ corresponding respectively to the classes from highest to lowest estimated probability mass, and labels for each of n examples in the dataset

1: **for** $i \in \{1, \cdots, n\}$ **do**
2: $L_i \leftarrow j$ such that $I_{i,j} = y_i$
3: $\hat{k}^* \leftarrow$ the $\lceil (1-\alpha)(1+n) \rceil$ largest value in $\{L_i\}_{i=1}^{n}$
4: **return** $\hat{k}^*$

Ensure: The estimate of the smallest fixed size set that achieves coverage, $\hat{k}^*$

Note: Adapted from [1] Appendix E

4.5 Contribution by members

- Adithya K Anil:
  1. Read the papers "A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification", "Tutorial on Conformal Prediction by Ryan Tibshirani," and "Uncertainty Sets for Image Classifiers using Conformal Prediction"
  2. Implemented the code for "Comparing Average Set Size vs α of two models" and did the analysis.
  3. Implemented the code for "Comparing Marginal coverage vs α of two models", "Deviation from True Marginal coverage vs α" and did the analysis.
  4. Implemented the code Quantile Regression, Conformalized Uncertainty Predictors, Jackknife, Jackknife+.
  5. Misc. logistics like report making, repository management etc.
- Rolla Siddharth Reddy:
  1. Read the papers "A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification", "Tutorial on Conformal Prediction by Ryan Tibshirani" and "Uncertainty Sets for Image Classifiers using Conformal Prediction"
  2. Implemented experiments for split conformal prediction - Analysis of multiple score function on conformal prediction.
  3. Implemented experiments for analysing dependence of size of calibration set on various parameters such as marginal coverage and predicted set sizes.
  4. Implemented multiple-sampling simulations for the experiments to model marginal coverage guarantee.
  5. Misc. logistics like report making, repository management etc.

- Nikhil Jamuda:
    1. Read the papers "A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification", "Tutorial on Conformal Prediction by Ryan Tibshirani" and "Uncertainty Sets for Image Classifiers using Conformal Prediction"
    2. Implemented the code for "RAPS vs Other Methods"
    3. Implemented Code for "Comparison of Different models using RAPS on Different dataset",
    4. Implemented "Finding Optimal λ for RAPS" (only λ part).
    5. Misc. logistics like report making, repository management etc.
- Pasupuleti Dhruv Shivkant:
    1. Read the papers "A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification", "Tutorial on Conformal Prediction by Ryan Tibshirani" and "Uncertainty Sets for Image Classifiers using Conformal Prediction"
    2. Implemented code for optimal k_{reg} hyperparameter selection.
    3. Implemented RAPS code on Imagenet-V2 demonstrating the *adaptive desideratum* as claimed by the authors ([1]).
    4. Misc. logistics like report making, repository management etc.