Conformal Prediction - Experiments

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

April 4th, 2025

Implementations

Comparison of Average Class Width and Marginal Coverage between two models with very distinct accuracies:

The two models considered in this analysis are *GaussianNB* and *LightGBM* classifier. Both are in-built models in scikit-learn.

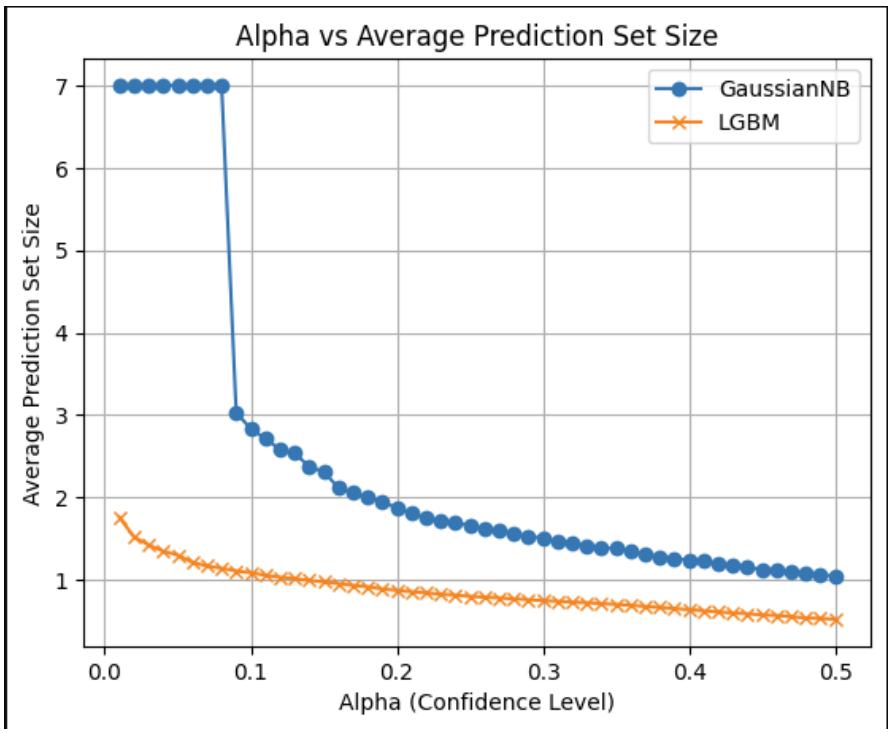
The dataset considered for this analysis is the "Covertype" dataset from the US Forest Service. It has a total of 581,012 samples, 54 features and 7 classes.

After training both the models on the same training data set and then measuring their accuracies on the test data set we get the following accuracies for each model.

- GaussianNB: 48.744 %
- Light GBM: 86.118 %

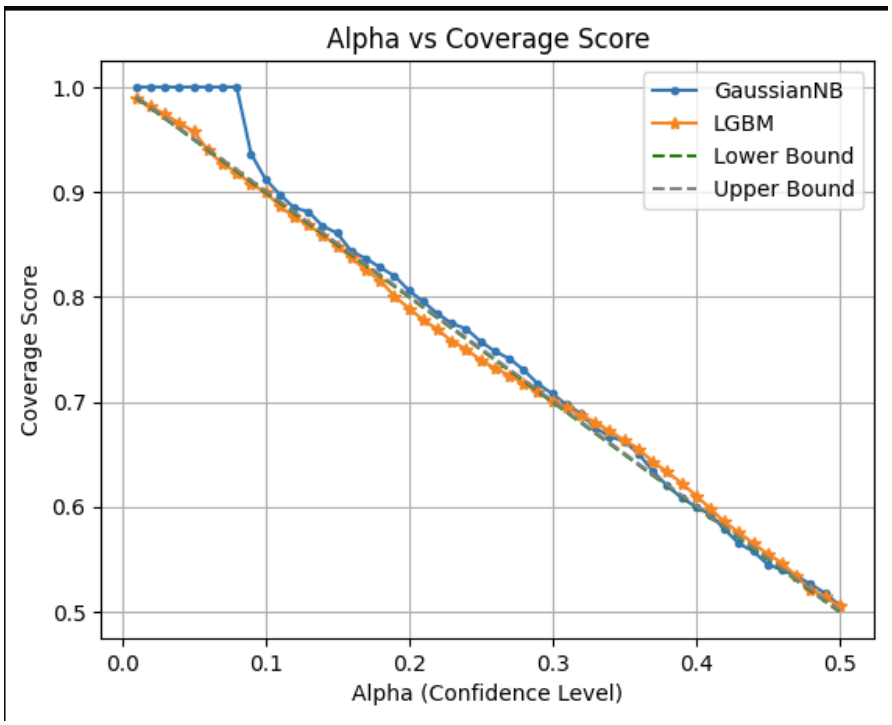
Clearly, GaussianNB is the poorer model compared to Light GBM. Now we attempt to create prediction sets for the test data set after creating scores and quantile predictor from the calibration data set.

To do this task, we have used **MAPIE** [**3**] library in python. The result obtained is the following plot:



Analysis:

- The average prediction set size of Gaussian NB is always more than the average prediction set size of 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 lesser 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:

- The marginal coverage of the Light GBM model is always very close to the lower bound $1-\alpha$ and the upper bound $1-\alpha+\frac{1}{n+1}$ however in some cases, exceeds the upper bound. This is because ties are present in some cases.
- The Gaussian NB model is always above the lower bound however it is always above the upper bound as well. This is because the model's accuracy is not very good so there are a lot of ties in the scores predicted of the calibration set.
- We see that the marginal coverage guarantee is not always guaranteed and this is because we have taken only one particular calibration data set and then predicted. The actual marginal coverage has to be measured over an average of all possible calibration data set and an infinite validation data set. (Ref: Gentle Introduction)

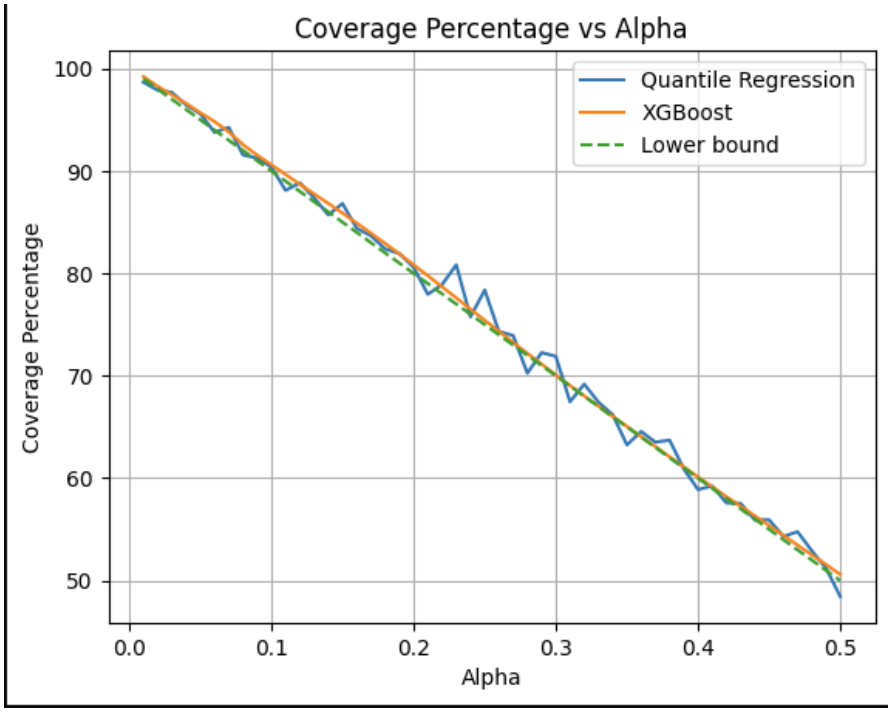
Comparison between Quantile Regression and Scalar Uncertainty Estimates:

To do this analysis, We have taken the California Housing Dataset. This data set has 20640 samples with 8 features. The prices are real values so we need to use a regression model.

Neural Networks are used to create quantile predictors in the quantile regression model and this runs fast($\approx 1s$) however on using neural networks to create a point predictor and an uncertainty predictor it takes about $\approx 46sec$ which is very slow.

To resolve this, we have used the XGBoost model and using in-built features in scikit-learn, we can easily find the standard deviation estimator. This takes about $1s$.

Now on plotting the results against α we get the following graph:



Analysis:

- The XGBoost model is much more smooth and consistent compared to the Quantile regression model which is much more variable.
- Both models tend to cross the lower bound at times which supports the claim made in *Implementation 1* that marginal coverage is not always guaranteed.
- In the "Gentle Introduction" paper, it was mentioned that quantile regression outperforms scalar uncertainty estimator usually because *there is no reason to believe that the uncertainty predictor is related to* α. However from the graph, we can infer that XGBoost is the better model since it doesn't drastically drop below the lower bound.

Prediction Set sizes

For different values of alpha we compute the distribution of set sizes of various randomly sampled test points. This gives us an idea of the models predictability power. Ideally we would expect or want a distributed size range such as the plot for $\alpha = 0.01$ of the LightGBM model. However we observe that for very small values of alpha this is satisfied for the better performing LightGBM model, however the GaussianNB model is consistently predicting sets of size 6 to satisfy coverage for such small alpha values. However on increasing alpha we see that the GaussianNB model starts to distribute the set sizes as expected. But in this range of alpha, the LightGBM model needs to predict a very small size of set since it had a good accuracy in the beginning itself.

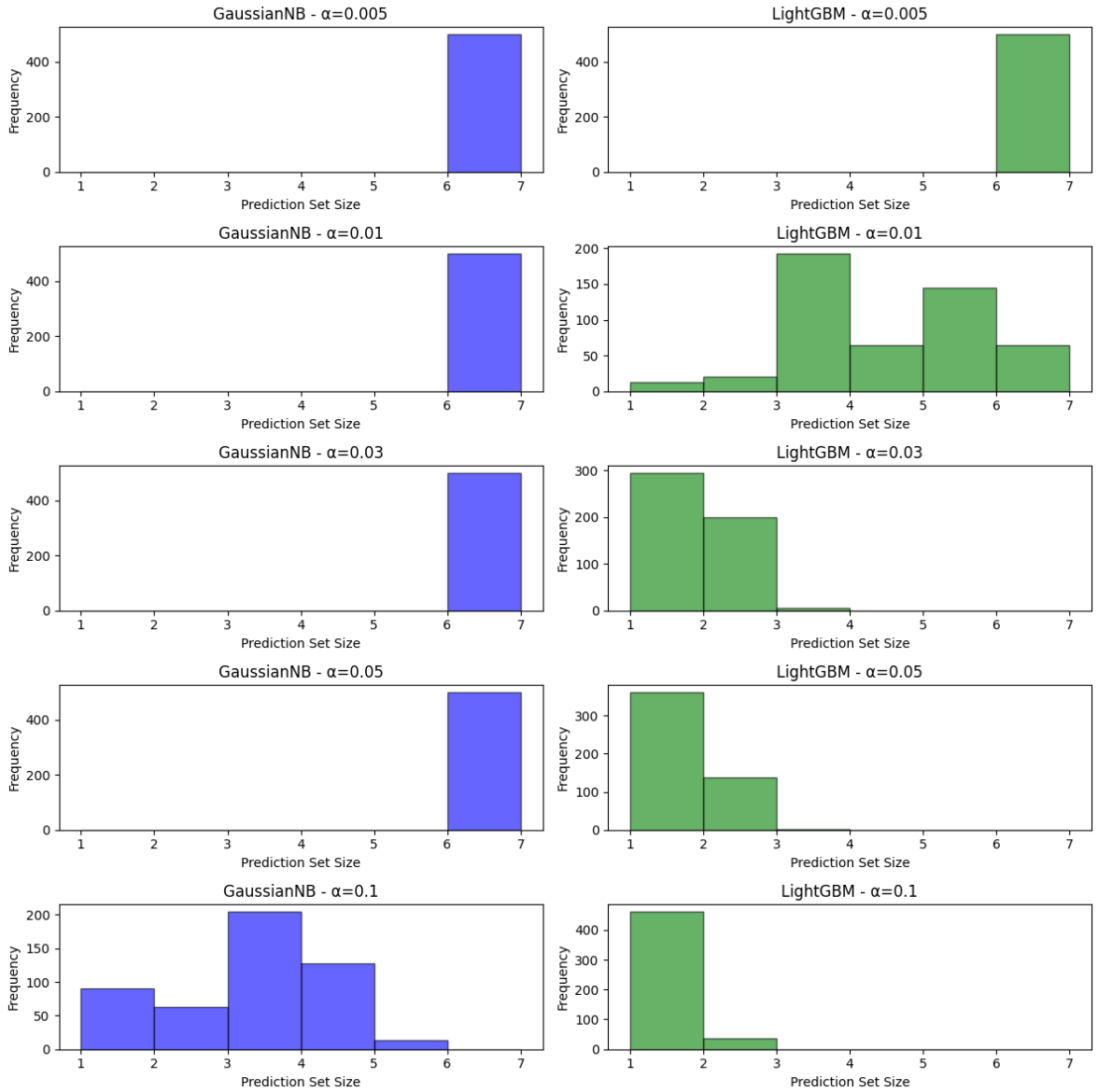


Figure 1: Prediction set sizes for randomly samples test points

Prediction This provides an empirical verification that the conformal prediction is working as expected for any type of model - thus making it model and distribution free.

Empirical Verification of effect of calibration sets

In this, we want to test if the coverage guarantee holds for any size of the calibration dataset. **Note :** In the analysis in *Sec 3.2* it was analyzed that the marginal coverage of the prediction sets is a random variable and according to [4], it should fit to a *Beta* distribution. Only over the randomness of the extracted calibration set will the coverage guarantee be satisfied.

$$P(Y_{\text{test}} \in C(X_{\text{test}}) \mid \{(X_i, Y_i)\}_{i=1}^{n}) \sim \text{Beta}(n+1-l, l),$$

where

$$l = \lfloor (n+1)\alpha \rfloor.$$

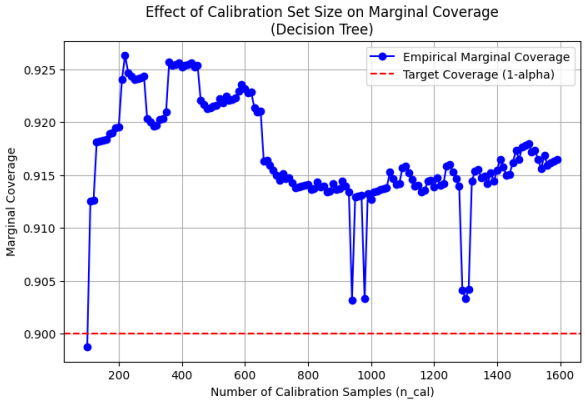


Figure 2: Decision Tree

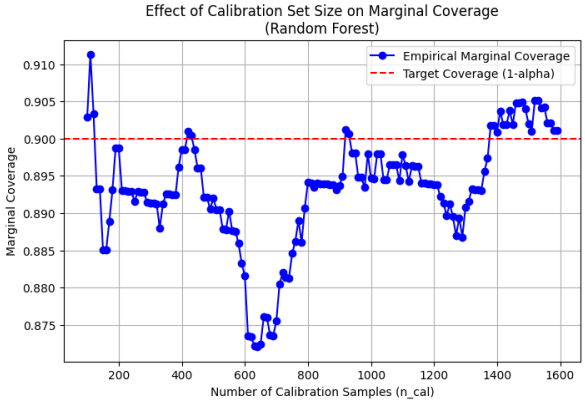


Figure 3: Random Forest

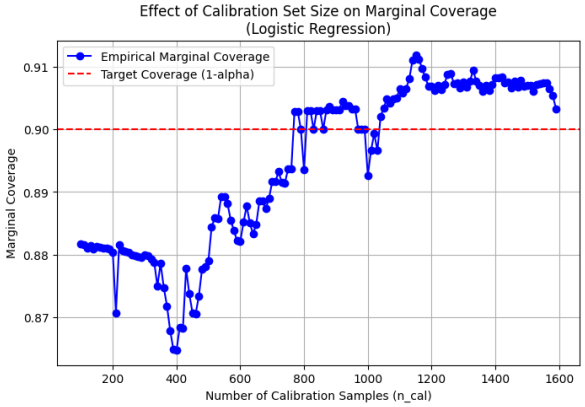


Figure 4: Logistic Regression

In these plots, we can see that the conditional guarantee is clearly not satisfied by every single calibration set. But over the randomness of this quantity, it turns out to be such that the conformal guarantee is still satisfied.

Beta distribution empirical evidence

We show that on an average

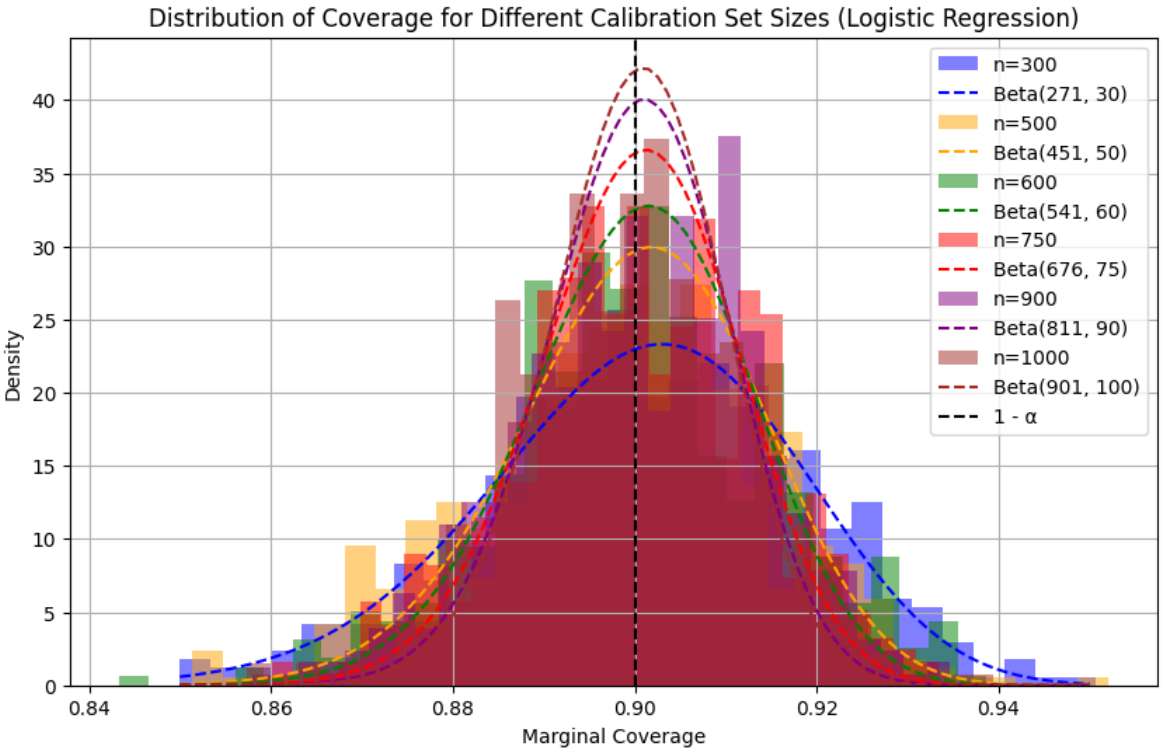
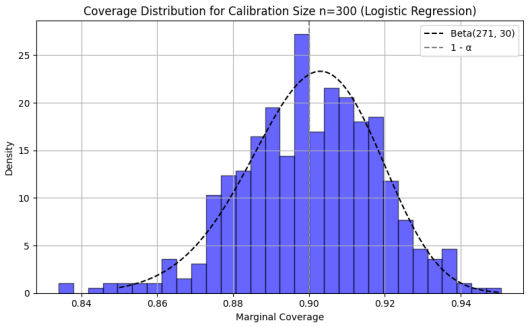
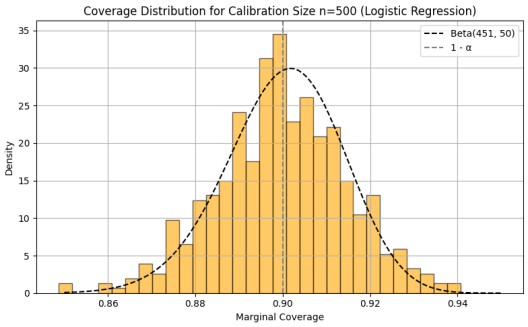


Figure 5: Effect of Calibration set size

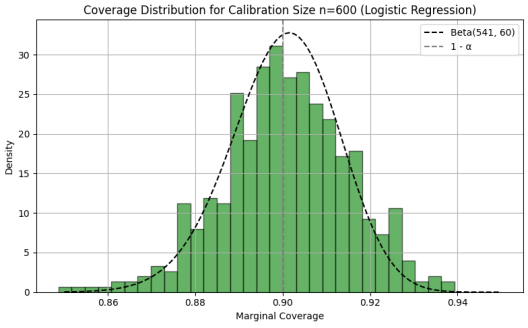
To observe each of the graph separately, we see that each of them roughly almost satisfies the beta distribution.



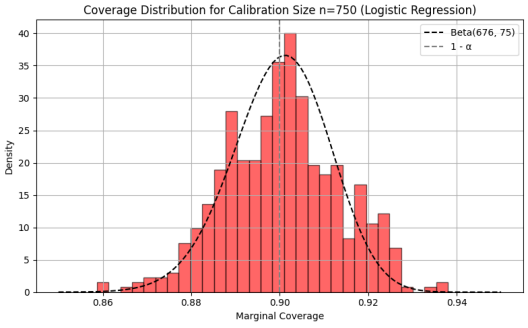
(a) Calibration size $n = 300$



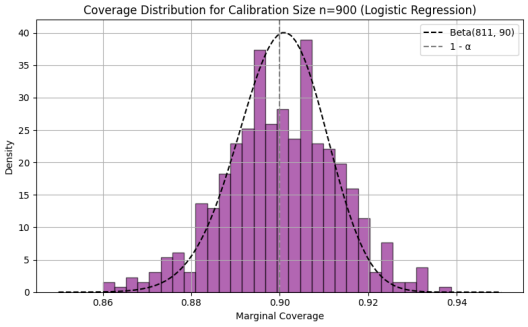
(b) Calibration size $n = 500$



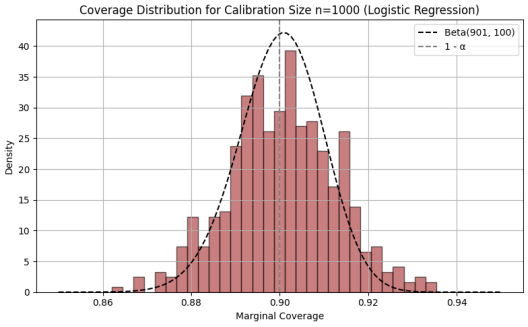
(c) Calibration size $n = 600$



(d) Calibration size $n = 750$



(e) Calibration size $n = 900$



(f) Calibration size $n = 1000$

Figure 6: Distribution of coverage for different calibration set sizes

Score Functions

In this section, we explore various (negatively-oriented) score functions and how they affect the marginal coverage guarantee for a bunch of models. We empirically test the coverage guarantees for these score functions.

The score functions used are :

- **1 -** $p_i(true - class)$
- **-**$log(p_i(x))$ - same x as above.
- **Entropy** - $\Sigma(-p_i.log(p_i))$
- **Top - 2 margin** : $p_{top-class}$ - $p_{second-top-class}$

Where each of the class probabilities are calculated by the softmax values of the models. Only models that can output softmax values are considered without any loss of generality.

The marginal scores for each of the implemented score functions are as follows :

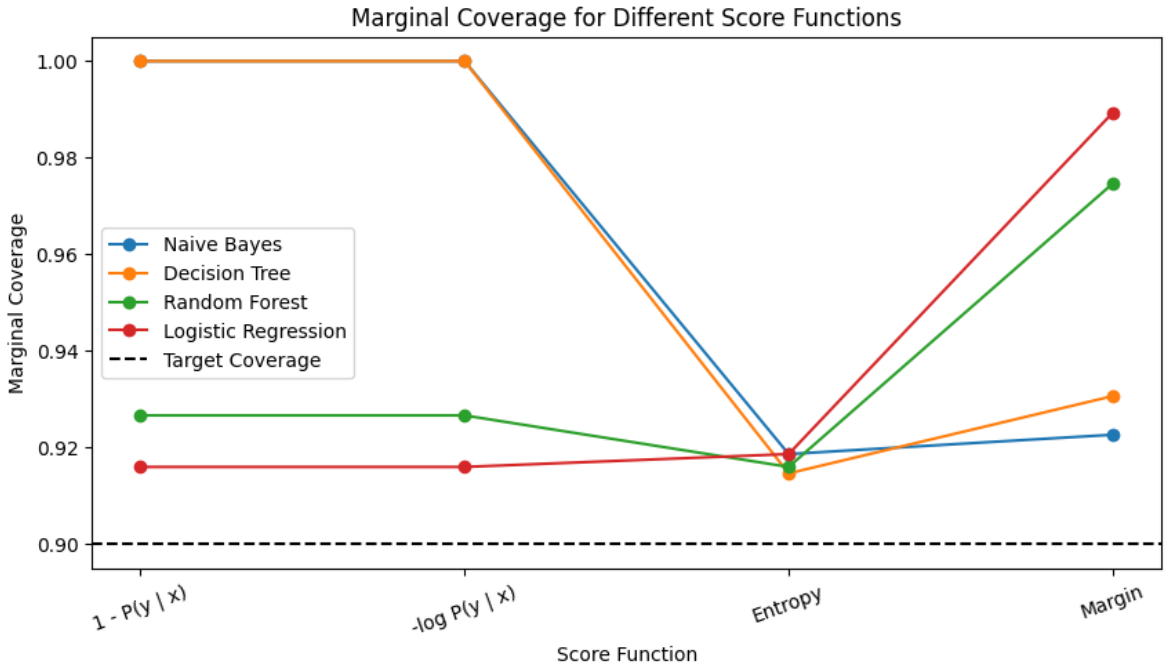


Figure 7: Marginal Coverage for the above mentioned score functions

We can clearly see that despite the choice of the score function here, they all satisfy the marginal coverage guarantee. This is a thing that we expected. When the parameters of the models were tuned to different values also a similar pattern was observed.

We also observe the differences in the average sizes of the predicted sets by the different models using the above mentioned score functions.

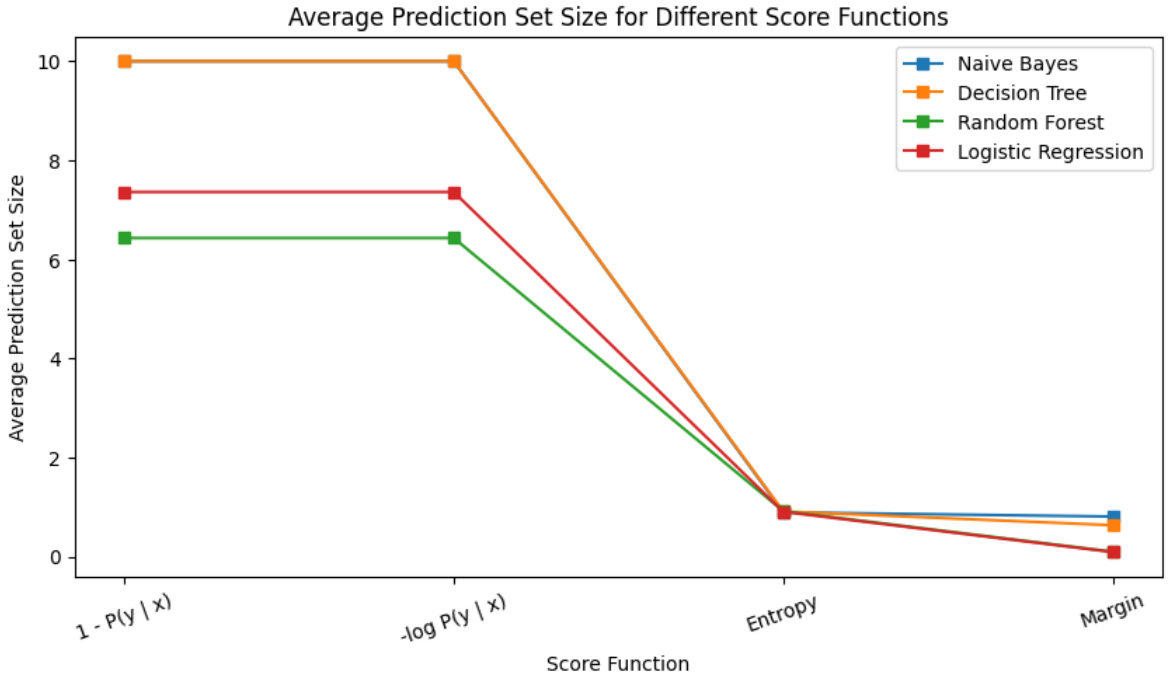
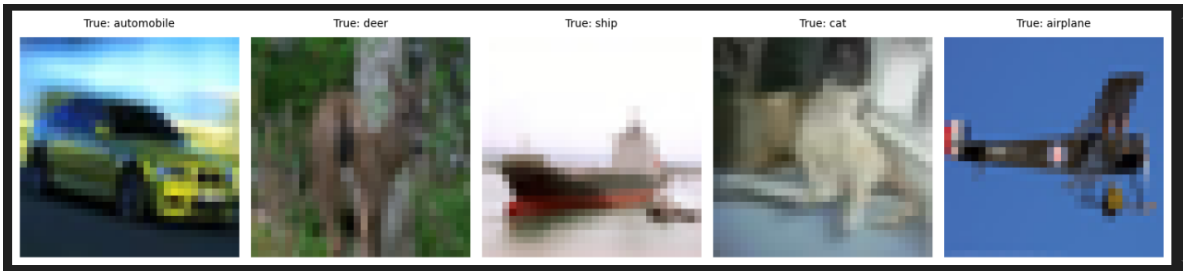


Figure 8: Average prediction set sizes for the models using the different score functions

Also some examples on real dataset - CIFAR - 10 (but compressed) :
A random sample from the test dataset:




```
===== 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 9: 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 10: Prediction sets on the CIFAR dataset by the logistic regression model

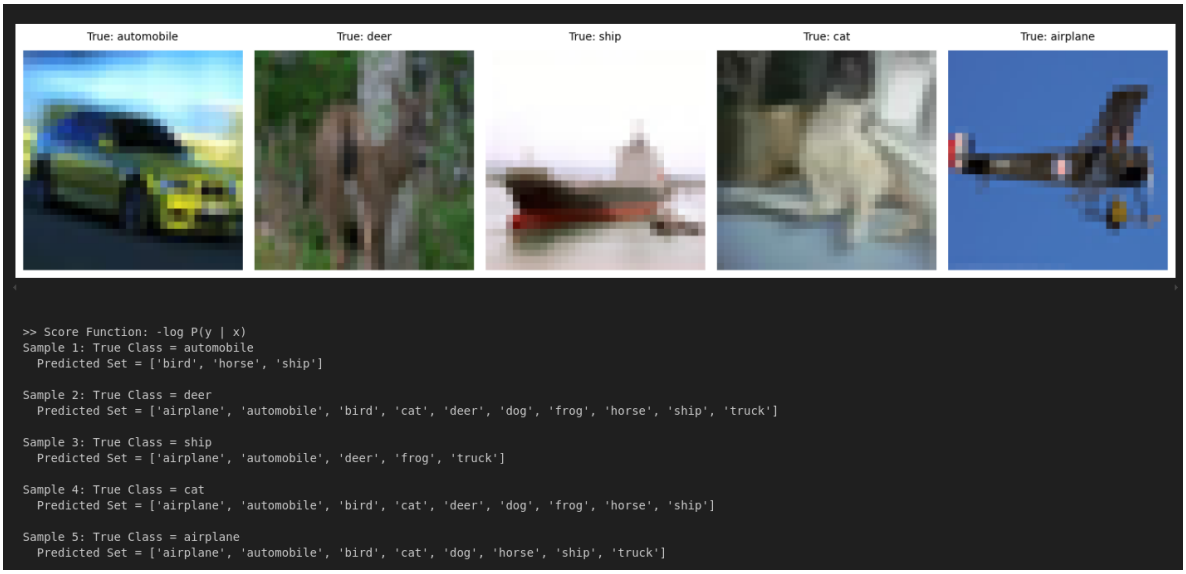


Figure 11: Score function used as negative log likelihood

Implementation of RAPS

In this section we implemented the RAPS (Regularized Adaptive Prediction Set) algorithm as mentioned in Angelopoulos' paper [2]. The code was implemented according to the algorithm described on the Project website [1].

The main objective of the implementation is to show that RAPS has less coverage than APS(Adaptive Prediction Set) with near the same coverage. The code was written without the `rand` variable given in the implementation of the paper [2].

We analyze the model `resnet-18` pre-trained version on the data set *Food101* with a split of 50k - 10k of training and test set. Further, the training was divided in 5000 data points as calibration set and 45000 as trainng set.

We got the following results for alpha values of [0.1, 0.15, 0.2, 0.25, 0.5]. With this we see the following results:

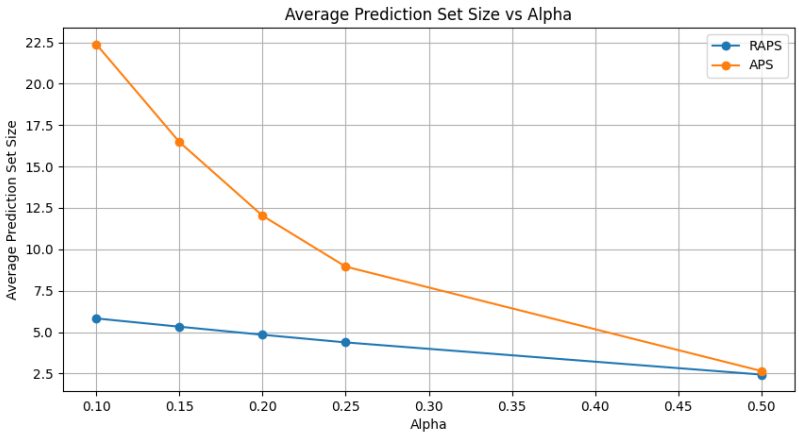


Figure 12: RAPS vs APS, Average prediciton set size

We can clearly see that the RAPS model has much lesser Prediciton set with the coverage (1- α)

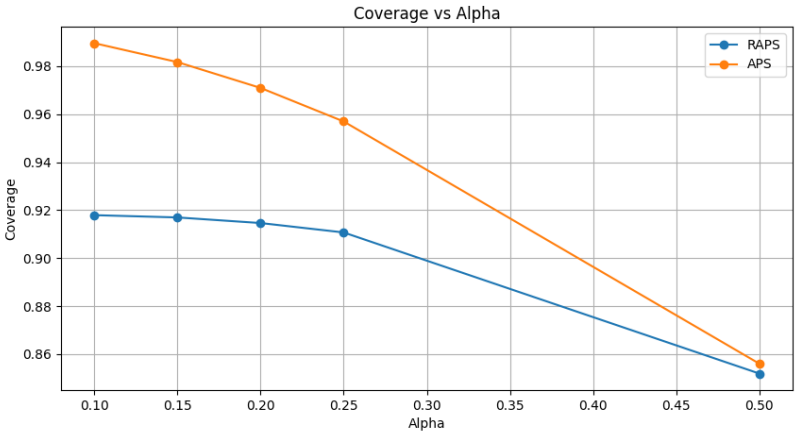


Figure 13: RAPS vs APS Average coverage size

as guarantee needed.

Jackknife and Jackknife+

As of now only the implementation has been done. Code can be found on the github repo.

References

[1] Anastasios N. Angelopoulos. Conformal classification. https://people.eecs.berkeley.edu/~angelopoulos/blog/posts/conformal-classification/, 2021. Accessed: 2025-04-07.

[2] 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.

[3] 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.

[4] Vladimir Vovk. Conditional validity of inductive conformal predictors, 2012.