TRUSTWORTHY AI MODELS

Training Report

BY YASH MITTAL, USAR, GGSIPU

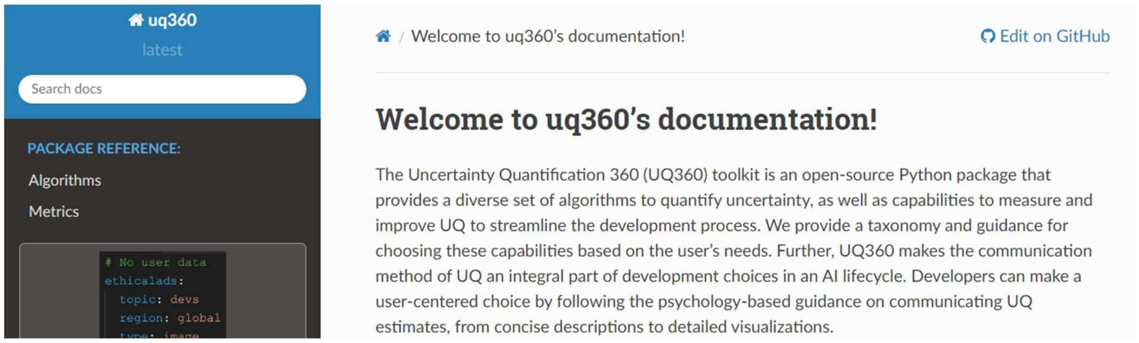
Index

Introduction

As artificial intelligence (AI) models continue to play an increasingly pivotal role in various industries, ensuring their trustworthiness becomes paramount, especially in the cases which involve life-threats. This report delves into a parameters of AI model trustworthiness, namely Uncertainty Quantification.

Uncertainty Quantification is a crucial aspect that involves assessing and managing uncertainty or doubtfulness associated with AI and ML models and their predictions. It also involves understanding the reliability and limitations of those predictions. We explored **Quantile Regression and BNN** and their Calibration in order to approximate and improve uncertainty measures of a model.

The IBM UQ360 toolkit is distinguished by its comprehensive and detailed approach. It delves deep into uncertainty quantification, encompassing a broad spectrum of uncertainty measures. It adopts various methodologies and employs diverse metrics to assess and quantify uncertainty. Notably, the toolkit also incorporates model recalibration techniques, further enhancing its robustness in handling uncertainty. This level of detail and flexibility makes it a valuable resource for advanced users and practitioners seeking an in-depth analysis of uncertainty in AI and ML models.



Welcome to uq360's documentation!

The Uncertainty Quantification 360 (UQ360) toolkit is an open-source Python package that provides a diverse set of algorithms to quantify uncertainty, as well as capabilities to measure and improve UQ to streamline the development process. We provide a taxonomy and guidance for choosing these capabilities based on the user's needs. Further, UQ360 makes the communication method of UQ an integral part of development choices in an AI lifecycle. Developers can make a user-centered choice by following the psychology-based guidance on communicating UQ estimates, from concise descriptions to detailed visualizations.

The report eventually underscores that the trustworthiness of AI models is not only a technical concern but also holds significant business and societal implications. The risks associated with deploying AI models that lack trustworthiness can lead to detrimental consequences, including biased decisions, regulatory non-compliance and reputational damage. As AI continue to shape industries and impact lives, the findings of this report emphasize the urgency of adopting strategies that prioritize and enhance the trustworthiness of AI models, thereby fostering responsible innovation and long-term success.

Uncertainty Quantification Estimation and Calibration – IBM UQ 360 Toolkit

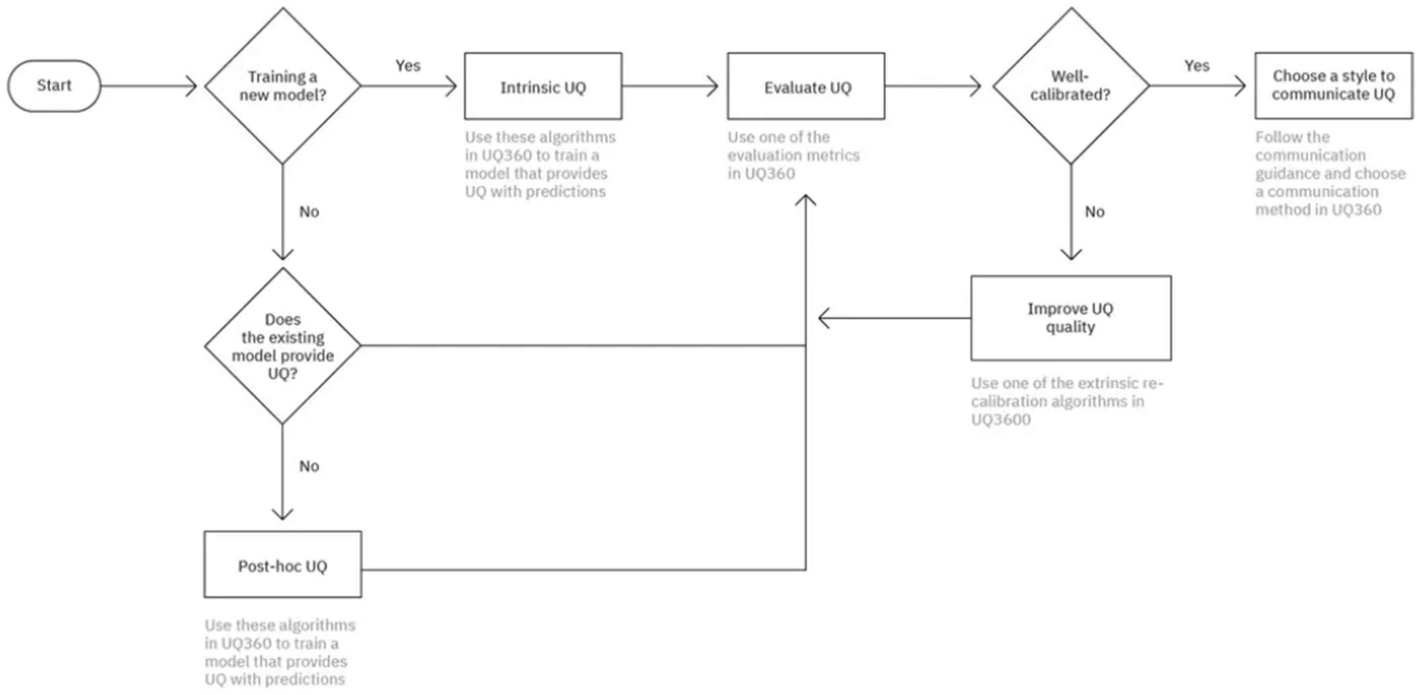
Uncertainty quantification (UQ) gives AI the ability to express that it is unsure, adding critical transparency for the safe deployment and use of AI.

Released at the 2021 IBM Data & AI Digital Developer Conference, The **Uncertainty Quantification 360 (UQ360) is an open-source toolkit** with a Python package to provide data science practitioners and developers access to state-of-the-art algorithms, to streamline the process of estimating, evaluating, improving, and communicating uncertainty of machine learning models as common practices for AI transparency.

Many models provide inherent uncertainties like a confidence or probability scores but there's more. UQ360 supports two major tasks into Uncertainty Quantification –

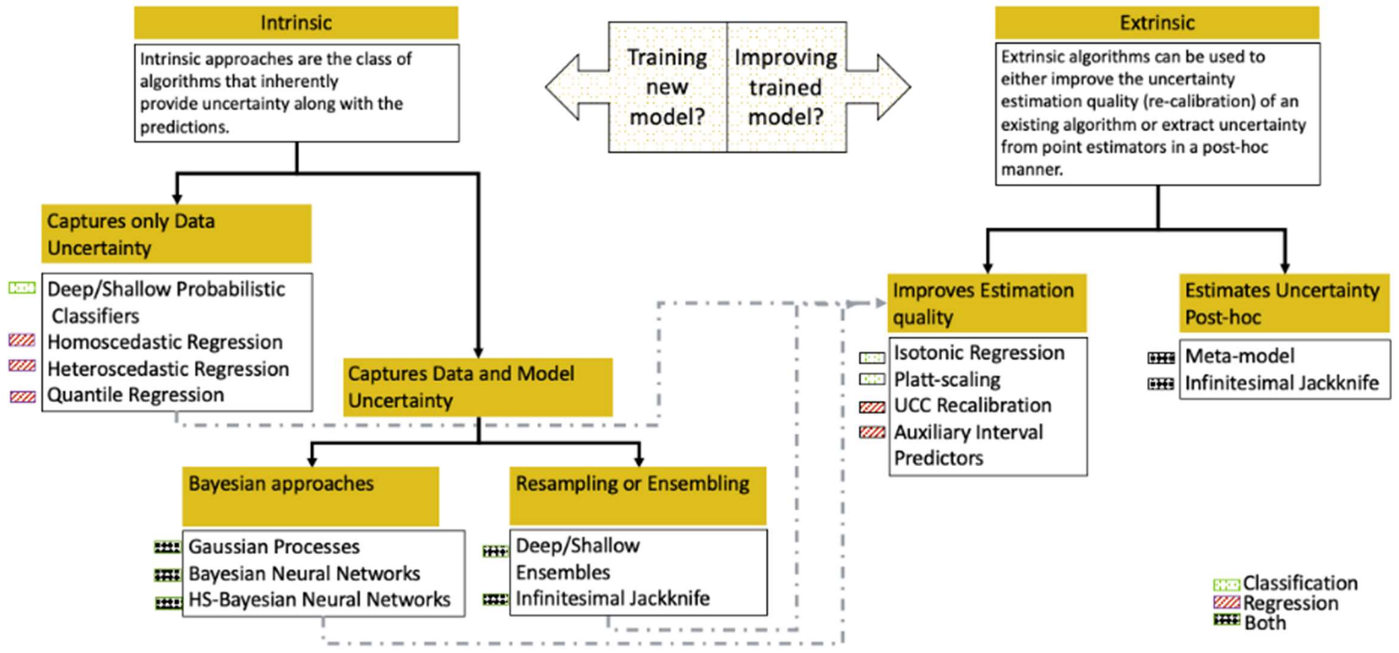
1. Provides any AI model the ability for AI model to say that it is unsure
2. Recalibration – Prevent models being Overconfident or Underconfident and ensuring reliable well-calibrated characteristics as per the metrics

Workflow



Start
Training a new model?
Yes
Intrinsic UQ
Use these algorithms in UQ360 to train a model that provides UQ with predictions
Evaluate UQ
Use one of the evaluation metrics in UQ360
Well-calibrated?
Yes
Choose a style to communicate UQ
Follow the communication guidance and choose a communication method in UQ360
No
Does the existing model provide UQ?
No
Improve UQ quality
Use one of the extrinsic re-calibration algorithms in UQ3600
No
Post-hoc UQ
Use these algorithms in UQ360 to train a model that provides UQ with predictions

UQ360 provides flexibility to add Uncertainty Quantification to an already existing AI model, or train a model from scratch which has UQ capabilities. The



Intrinsic
Intrinsic approaches are the class of algorithms that inherently provide uncertainty along with the predictions.
Training new model?
Improving trained model?
Extrinsic
Extrinsic algorithms can be used to either improve the uncertainty estimation quality (re-calibration) of an existing algorithm or extract uncertainty from point estimators in a post-hoc manner.
Captures only Data Uncertainty
Deep/Shallow Probabilistic Classifiers
Homoscedastic Regression
Heteroscedastic Regression
Quantile Regression
Captures Data and Model Uncertainty
Bayesian approaches
Gaussian Processes
Bayesian Neural Networks
HS-Bayesian Neural Networks
Resampling or Ensembling
Deep/Shallow Ensembles
Infinitesimal Jackknife
Improves Estimation quality
Isotonic Regression
Platt-scaling
UCC Recalibration
Auxiliary Interval Predictors
Estimates Uncertainty Post-hoc
Meta-model
Infinitesimal Jackknife
Classification
Regression
Both

former is accomplished with the help of **intrinsic approaches** and the latter with the **extrinsic approaches**. The trained model with UQ capabilities is evaluated through **appropriate metrics** as defined or further with Uncertainty Characteristics Curve. If the metrics stand good and model looks **well-calibrated**, they are **communicated with the stakeholders** maybe with verbal notations of Good or Bad, Numerical values, or Density plot to see a greater analysis. Otherwise, **re-calibration** is done using the various algorithms in extrinsic approaches until the model seems well-calibrated and thereby communicated as needed.

Types of Uncertainty

Aleatoric (Data) Uncertainty –

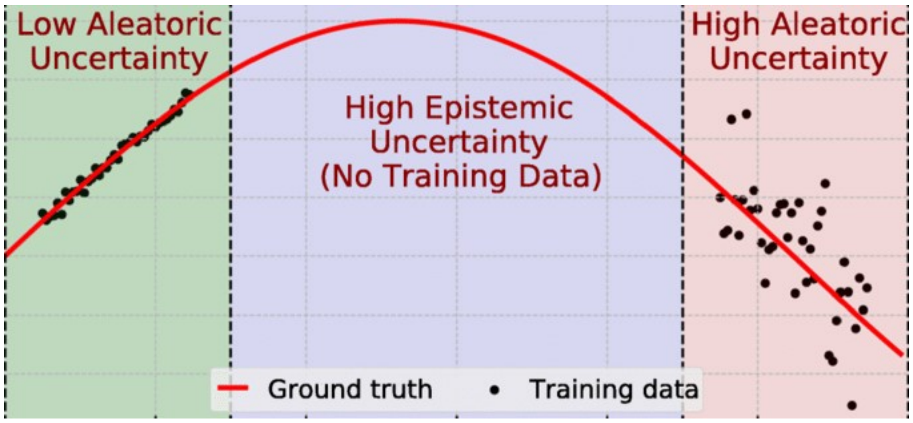
Random fluctuations in between the input features and the outcome in the training data, also referred to as intrinsic noise. It occurs when near similar parameter values show big difference in output as a result of missing covariates. It mainly arises from data generating process and one way to reduce data uncertainty is by adding new features that could account for the fluctuations. For example, adding neighbouring commodities details in the housing price prediction. But it's not always reducible easily.

This uncertainty can be visualised by a simple scenario. Suppose we want to shoot an arrow and have all the data, distance, angle, material, target details and others needed. Now, <u>if we shoot the arrow twice keeping all the parameters same, it will still hit the target with a slight noise, at a slightly moved point</u>. This can be thought of due to wind or maybe some more variates that just can't be taken directly into account.

Epistemic (Model) Uncertainty –

It refers to the lack of knowledge about an underlying process of the model fitted. It states that the model is not trained well and not seen much data of that type or process. It can be the case that for some interval the model is quite sure but would be unsure if we extrapolate the test instance to a very-high interval. It is a kind of 'Reducible' kind of uncertainty and generally, can be mitigated by collecting more training data to improve the fitting and filter out alternative model configurations.

For example, if we train a model for house price predictions on 1-10 rooms. The data could be good for 1-5 rooms but less for 5-10 rooms. If we test for an instance of 8 rooms, the model will be quite uncertain and if we test for an instance of 26 rooms, the model will be highly uncertain.



Regression UQ

The dataset used in this research is the Boston housing dataset which contains information about various housing-related factors in different neighbourhoods in Boston, Massachusetts.

The columns and features of this dataset are as follows-

- CRIM: Per capita crime rate by town.
- ZN: Proportion of residential land zoned for large lots.
- INDUS: Proportion of non-retail business acres per town.
- CHAS: Charles River dummy variable (1 if the tract bounds the river, 0 otherwise).
- NOX: Nitrogen oxide concentration (parts per 10 million).
- RM: Average number of rooms per dwelling.
- AGE: Proportion of owner-occupied units built before 1940.
- DIS: Weighted distance to employment centres.
- RAD: Index of accessibility to radial highways.
- TAX: Property tax rate.
- PTRATIO: Pupil-teacher ratio by town.
- B: Proportion of residents of African American descent.
- LSTAT: Percentage of lower-status population.

We choose to take 4 features namely **‘CRIM’, ‘RM’, ‘PTRATIO’, ‘DIS’** from the overall set. The target variable in this dataset is the median value of owner-occupied homes (in thousands of dollars), which is denoted as **‘MEDV’**. The dataset therefore has **506 rows and 5 columns including target**.

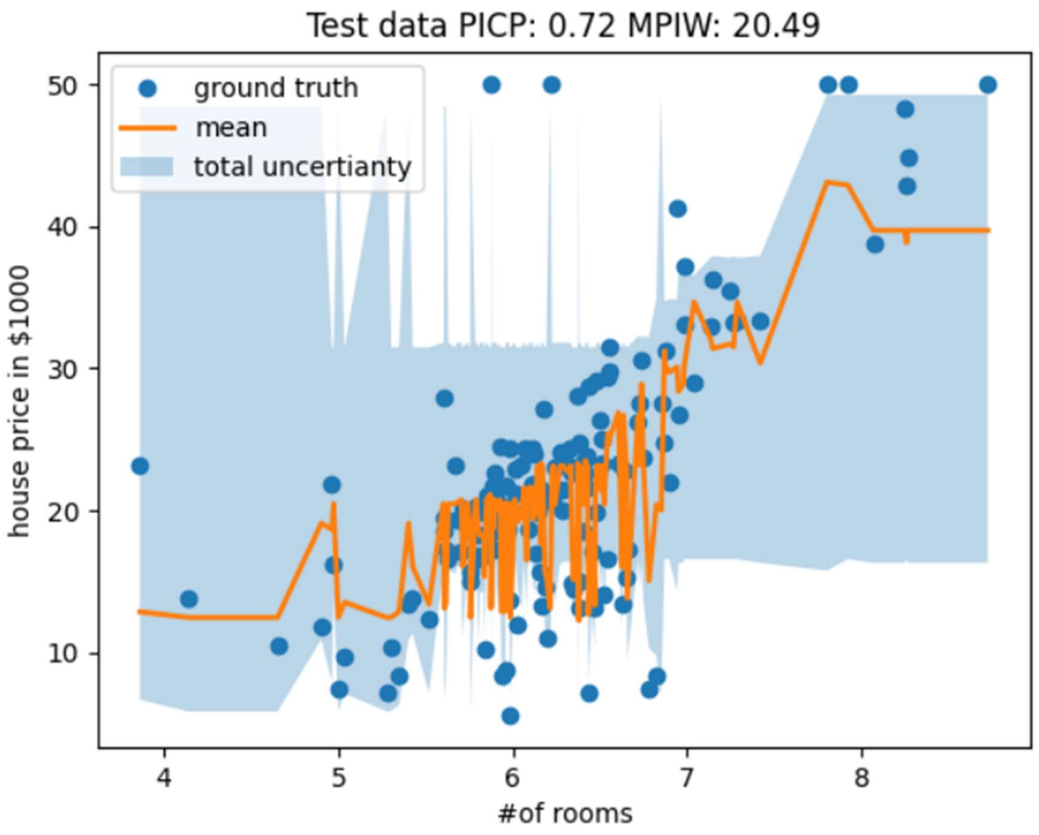
We fitted 1-intrinsic regression model as **Quantile Regression**, and 1-extrinsic regression model as **Auxiliary Interval Predictor**, from the UQ360 toolkit to analyse uncertainty measure and compare the two.

We imported all necessary libraries, the dataset, scaled it and divided data into 70-30 split while preprocessing for the input. The Quantile Regression was given the configuration parameters as alpha:0.95, n_estimators:20, max_depth:3, learning_rate:0.1, min_samples_leaf:20, min_samples_split:20.

The prediction is evaluated mainly on the metrics **Prediction Interval Coverage Probability (PICP) and Mean Prediction Interval Width (MPIW).**

PICP – It is defined as the number of intervals which actually contained the ground truth divided by all the prediction intervals. We got to maximise this.

MPIW – It is mean of the bandwidth of all the prediction intervals. We got to minimize this, large interval brings unsurety.



The regression metrics values come out to be -

'RMSE': 5.95275338972838, #root-mean-square error

'NLL': 2.9116479017158396, #negative-log-likelihood

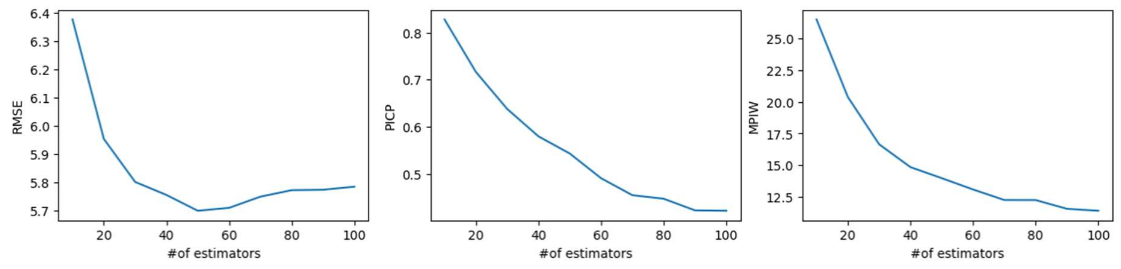
'AUUCC_gain': -0.23644846101969086, #area under uncertainty char. curve

'PICP': 0.881578947368421,

'MPIW': 20.492301889695252,

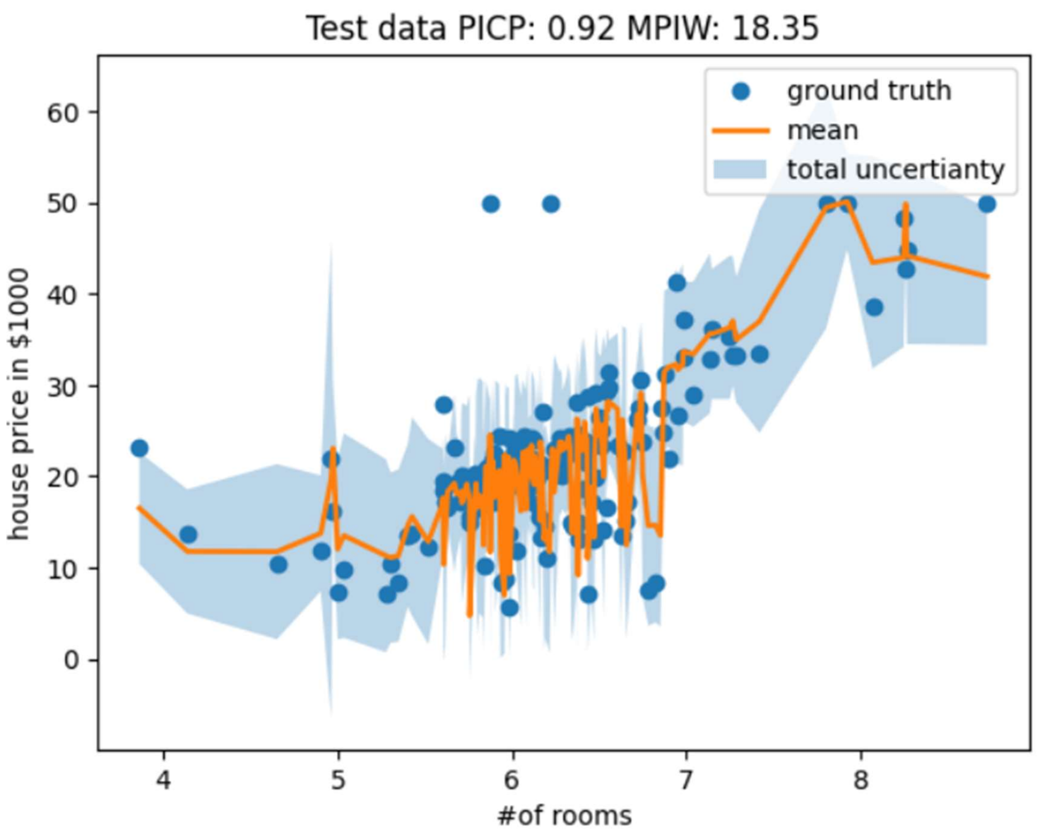
'R2': 0.5744303196525262 #R2-Score

We then evaluate model on metrics of RMSE, PICP, MPIW as no. of estimators are changed –



Both of PICP and MICP declines with the increase in estimators. While the RMSE score declines rapidly at the start and increase slightly after 50 estimators.

Next, we train an extrinsic model, Auxiliary Interval Predictor. We don't take any model as base model and the model_type - mlp (the only supported currently). The configuration parameters are given the values, "num_features": X_train.shape[1], "num_outputs": y_train.shape[1], "num_hidden": 50, "batch_size": 32, "lr": 0.01, "num_main_iters": 50, "num_aux_iters": 5, "num_outer_iters": 5, "lambda_noise": 0.001, "lambda_sharpness": 0.01, "lambda_match": 1.0, "calibration_alpha": 0.90.



Evaluating on same regression metrics values here come out to be –

'RMSE': 5.9181885081564385

'NLL': 3.127437549647439

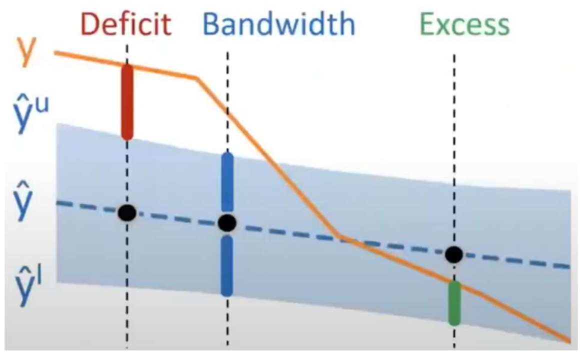
'AUUCC_gain': -0.01960630810715826

'PICP': 0.9210526315789473

'MPIW': 18.346758

'R2': 0.5793581432138188

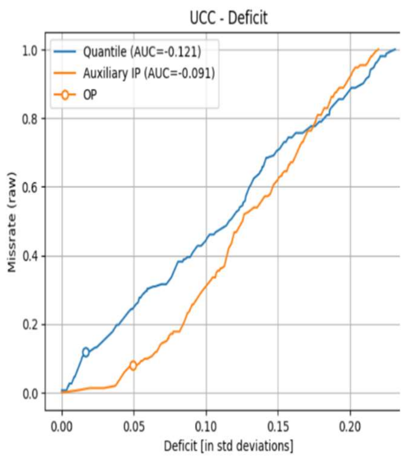
Further, we analyse the performance of models on a powerful feature of IBM UQ360, that is, **Uncertainty Characteristics Curve (UCC).**It allows us to plot

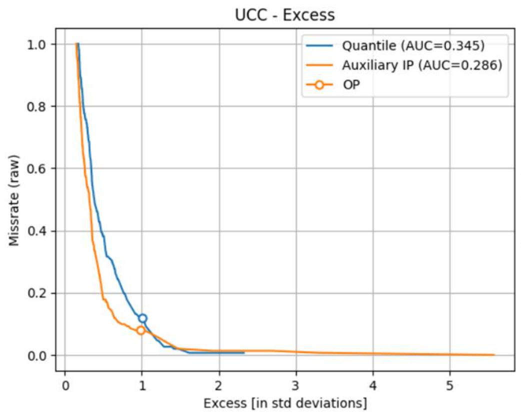


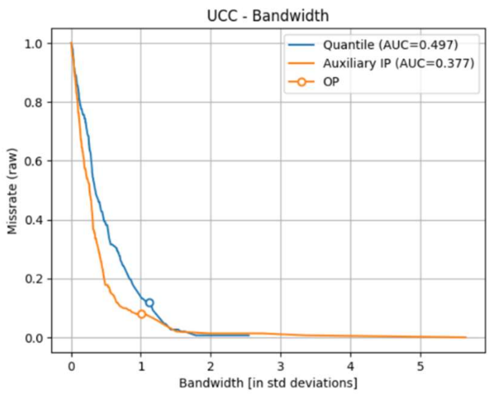
bi-dimensional graphs with any of the allowed four metrics for the particular samples.

1. Missrate – It is 1 – PICP (no. of times missed ground truth/all samples)
2. Excess – Amount of interval that's extra after reaching ground truth
3. Deficit – Amount of interval needed more to reach ground truth
4. Bandwidth – Interval width of a sample

We plotted the missrate with the other 3 parameters –

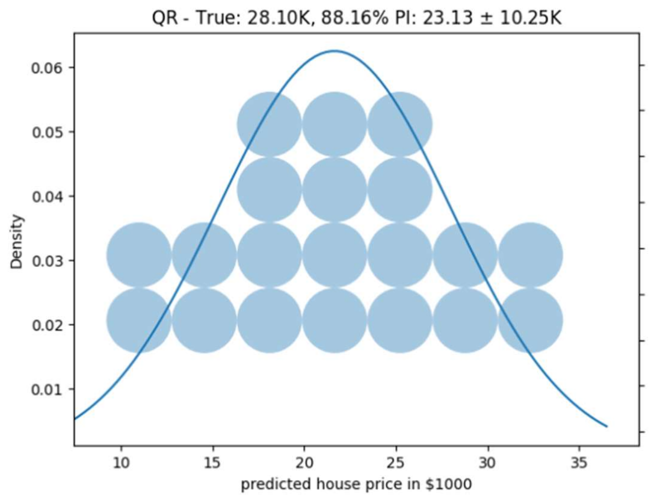


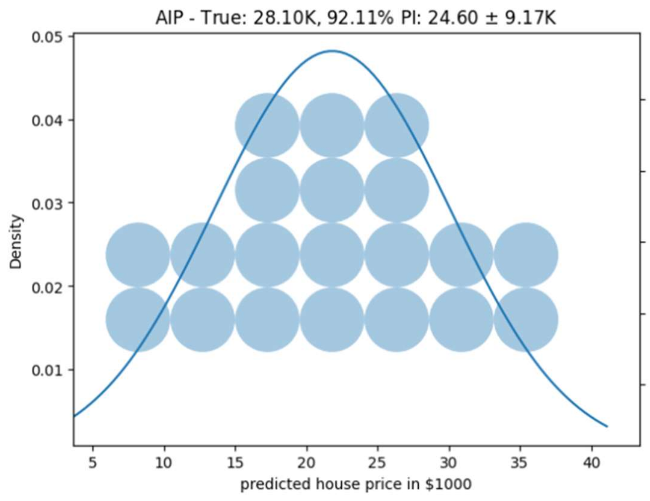




From the PICP and MPIW, we might interpret that AIP is better than QR. But with the UCC curves, we can see that **AIP performs better when the bandwidths, excess and deficit are small**, whereas **QR performs better at some regions where deficit, excess, and bandwidths get large enough**.

The final step is to **communicate the predictions** with their uncertainties. The output in regression cases is a prediction interval with a mean and side bound.





We choose to communicate a single instance test_data[100]. True value, Confidence, and Prediction Interval are depicted numerically in title of the plot for a simple and easy to interpret view to stakeholders. Whereas the graph shows a greater depth into the distribution for the technical chairpersons.

Classification UQ

The dataset used in this research is the Adult Income dataset, a multivariate dataset that was extracted from the 1994 United States Census database. It contains 48,842 records and 14 attributes. The dataset is commonly used for classification tasks, especially for predicting an individual's income level (>50K or <=50K), based on their demographic features. It is a balanced dataset, with approx. 24,000 records for each income category. The attributes are mostly categorical, with a few continuous attributes.

The attributes –

- Age: Age of the individual.
- Workclass: The type of employment, such as Private, Self-emp-not-inc, Local-gov, etc.
- Education: Highest level of education achieved.
- Education-Num: Numeric representation of education level.
- Marital Status: Marital status of the individual.
- Occupation: Type of occupation.
- Relationship: Relationship status (e.g., Husband, Wife, Not-in-family).

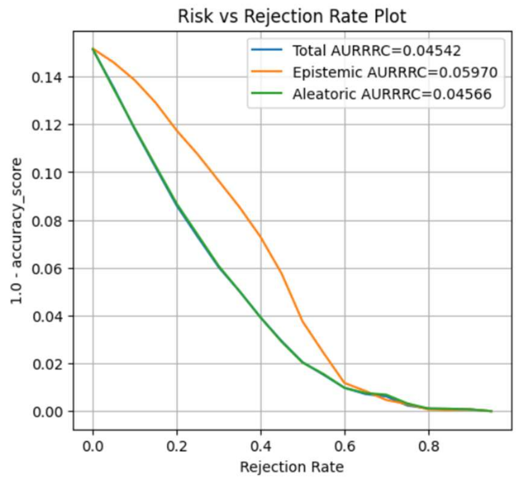
- Race: Ethnicity or race of the individual.
- Sex: Gender of the individual.
- Capital Gain: Income from investment sources, apart from salary/wages.
- Capital Loss: Losses from investment sources, apart from salary/wages.
- Hours per week: Number of hours worked per week.
- Native Country: Country of origin.

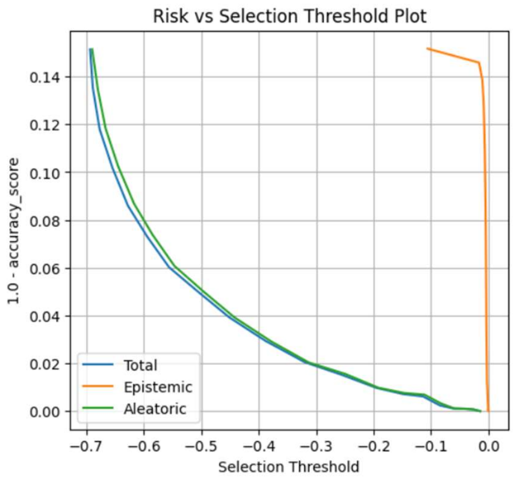
To train our model here for UQ, we use **Gaussian-based Bayesian Neural Network (BNN) Intrinsic UQ Model.** Bayesian neural network (BNN) is a type of neural network that uses Bayesian inference to learn the network parameters. Bayesian inference is a statistical framework that allows us to quantify the uncertainty in our estimates. BNNs are more robust to overfitting than traditional neural networks. Gaussian-based Bayesian Neural Networks offer a probabilistic approach to neural network modeling, allowing for the estimation of uncertainty in predictions. GBNNs uses a Gaussian prior distribution to model the uncertainty in the network parameters. The Gaussian prior distribution is a conjugate prior to the Gaussian likelihood function, so that it is easily incorporated into the Bayesian inference procedure.

The dataset was pre-processed appropriately with multi indexing, One-hot encoding, scaling, imputing, feature engineering. Subsequently, the BNN model was trained with the following configurations,

"ip_dim": X_train.values.shape[1], "op_dim": 2, "num_nodes": 128, "num_layers": 1, "num_epochs": 10, "step_size": 0.001.

We predict and transform results in a way to separate out Aleatoric and Epistemic Uncertainty and plot a **metric - risk vs rejection rate curve**.

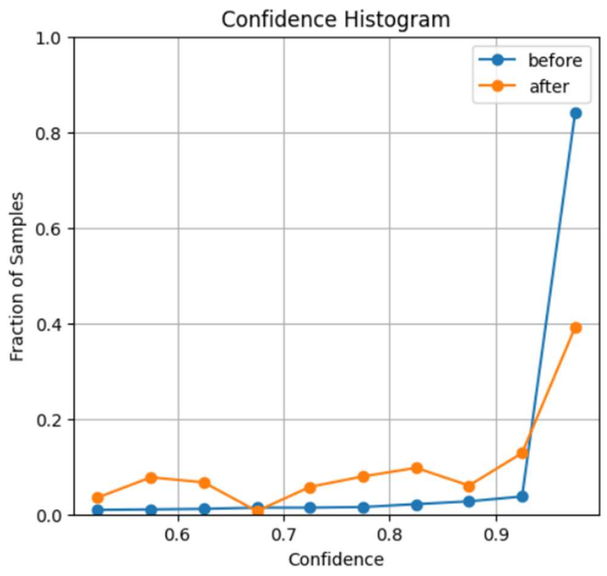


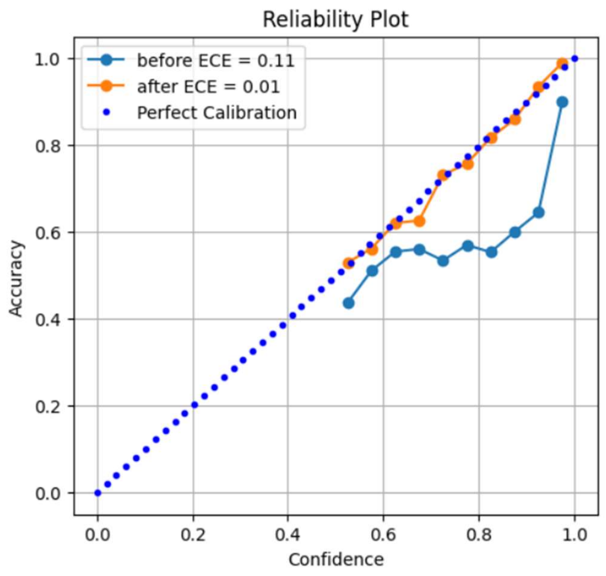


The uncertainty increases quite similarly with the rejection rate but the Epistemic Uncertainty seems to be a bit more in the model. Also, the aleatoric uncertainty seems to drop at a faster rate than epistemic through the curve on right. These results indicate that the model needs calibration.

We calibrate the model using **Classification_Calibration Extrinsic Model** on our base BNN model and predict using it.

Now we use another **metric – Reliability Diagram** to check and compare before and after Calibration results. Reliability plot and Confidence Histogram gives us a measure of uncertainty and accuracy of a model to determine that the underlying model stands up to its confidence score. If a model has right confidence, then over all the samples of that confidence score it should show that amount of accuracy (a line to 45degree). Otherwise, the model is underconfident or overconfident.





In this case, the model was over-confident to many samples earlier before calibration but fitted well to the shown confidence after calibration (near to perfect calibration curve).

Conclusion and Key Findings

- Many models give uncertainty outlook in a simple direct way such as a probability or confidence score through a **SoftMax layer, but there's much more** to Uncertainty Quantification.

- It is necessary to check on trustworthiness of AI models and the parameters we explored play a vital role. **We must have explainability to the Whitebox or Blackbox models** which we use to predict and they **must not be biased** towards any subgroup while deployed and working in a real-world scenario which may lead to discrimination. Moreover, Uncertainty Quantification helps to take better decisions over model predictions, an Apple vendor can get the approximate interval of demand and stock back with appropriate interval negotiations to the price trade-off.

- UQ360 is an extensive toolkit by IBM to quantify uncertainty in a **wide range of models and datasets**. **It only supports Python3.7** and need to setup workspace environment appropriately to that by any means. It has quite a number of possible algorithms, from which one can choose based on model and data, to pick out the Uncertainties along with calibrating and communicating them. The complete documentation and source code is available open-source and can be used to play for new experiments. Uncertainty Characteristics Curve, is a powerful tool supported by only UQ360. Requires fairly high knowledge of ML and Statistics to take the most out of it, but the toolkit offers in depth view to uncertainty and can prove to be a good method to quantify uncertainty in a model.

- ## Future Scope

  - Exploring other notable and powerful algorithms in UQ360, such as Gaussian Process Regression, MetaModeling, Monte-Carlo Dropout, and Infinitesimal Jackknife, Neural Ensemble Methods
  - CNN model for UQ360
  - Communicating aggregates of samples by various means in the UQ360
  - Experimenting with model blends, such as a recent paper came which did Monte-Carlo Dropouts and treated each of it as an ensemble model, combining the best of two.

References

- https://www.coursera.org/specializations/deep-learning
- https://aiverifyfoundation.sg/
- https://github.com/IMDA-BTG/aiverify
- YouTube - Explainable AI explained! | #4 SHAP by DeepFindr
- https://imda-btg.github.io/aiverify/
- https://imda-btg.github.io/aiverify-developer-tools/getting_started/start_here/
- Keynote: Fairness of Machine Learning in Medical Image Analysis - Enzo Ferrante | SciPy 2022
- SSA 5 | Aequitas Tutorial
- Uncertainty Quantification 360: A Hands-on Tutorial | PyData Global 2021
- https://uq360.res.ibm.com/
- https://uq360.readthedocs.io/en/latest/
- https://github.com/IBM/UQ360
- Florian Wilhelm: Are you sure about that?! Uncertainty Quantification in AI | PyData Berlin 2019
- Quantile Regression as The Most Useful Alternative for Ordinary Linear Regression
- Massimiliano Ungheretti- Modelling the extreme using Quantile Regression| PyData Global 2020
- Uncertainty estimation and Bayesian Neural Networks - Marcin Możejko
- Uncertainty Prediction for Deep Sequential Regression Using Meta Models
- https://oecd.ai/en/catalogue/tools