Probability Calibration in Binary Classification Models: An Overview with Python Examples

Charlie Li · Following

3 min read · Apr 9

Introduction

As data scientists, we often attach great importance to ensuring that predicted probabilities from machine learning models are well-calibrated. A well-calibrated model is critical for many applications in the real world, such as medical diagnosis, fraud detection, and probability of default prediction. If a model is poorly calibrated, it may assign probabilities that are too extreme, leading to poor decision-making. For example, if a model assigns a default probability of 0.6 to a loan contract, while the real probability of such a loan is only 0.3, then the loan issuer may overprice or reject the loan application.

What is Probability Calibration?

Probability calibration is the process of adjusting the model output probabilities to reflect the true probabilities seen in the data. Calibration is especially important for binary classification models because these models assign a probability of either 0 or 1 to each input, which is then used to determine the final classification. Ideally, these probabilities should be well-calibrated so that the model's predictions are accurate.

Methods of Probability Calibration

There are several methods of probability calibration, including Platt scaling and isotonic regression.

Platt scaling is a parametric method that fits a logistic regression model to the predicted probabilities and uses this model to transform the probabilities.

```
from sklearn.calibration import CalibratedClassifierCV
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier

# Create an uncalibrated classifier
clf = RandomForestClassifier()

# Fit the classifier to your data
clf.fit(X_train, y_train)

# Calibrate the probabilities using Platt scaling
platt_clf = CalibratedClassifierCV(
clf, method='sigmoid', cv='prefit', base_estimator=LogisticRegression()
)
platt_clf.fit(X_train, y_train)

# Use the calibrated classifier to make predictions
y_calibrated_pred = platt_clf.predict(X_test)
```

Isotonic regression is a non-parametric method that fits a piecewise-constant function (non-decreasing) to the predicted probabilities and adjusts the probabilities based on this function. Isotonic regression tends to perform better than Platt scaling, especially when the original model has highly non-linear decision boundaries.

```
from sklearn.isotonic import IsotonicRegression
from sklearn.ensemble import RandomForestClassifier

# Create an uncalibrated classifier
clf = RandomForestClassifier()

# Fit the classifier to your data
clf.fit(X_train, y_train)

# Calibrate the probabilities using isotonic regression
iso_reg = IsotonicRegression()
calibrated_probs = iso_reg.fit_transform(clf.predict_proba(X_test)[:, 1])

# Use the calibrated probabilities to make predictions
y_calibrated_pred = (calibrated_probs >= 0.5).astype(int)
```

How do we evaluate if the model is well-calibrated?

There are several methods for evaluating the calibration of a binary classification model, including calibration plots, calibration error, and the Brier score.

One common method for visualizing the calibration of a binary classification model is to use a calibration plot. A calibration plot compares the predicted probabilities with the true probabilities of the positive class by dividing the predicted probabilities into bins and plotting the average predicted probability and the true frequency of the positive class in each bin.

A well-calibrated model should have a calibration plot in which the predicted probabilities closely match the true probabilities.

```
import matplotlib.pyplot as plt
from sklearn.calibration import calibration_curve

# Generate calibration plot
prob_true, prob_pred = calibration_curve(y_true, y_pred, n_bins=10)
plt.plot(prob_pred, prob_true, marker='o')
plt.plot([0, 1], [0, 1], linestyle='--')
plt.xlabel('Predicted Probability')
plt.ylabel('True Probability')
plt.title('Calibration Plot')
```

The Brier score is another common metric for evaluating the calibration of a binary classification model. The Brier score is calculated as the mean squared difference between the predicted probabilities and the true binary outcomes. A well-calibrated model should have a low Brier score, indicating that the predicted probabilities are well-calibrated and closely match the true probabilities.

```
from sklearn.metrics import brier_score_loss

# y_true: true binary labels (0 or 1)
# y_prob: predicted probabilities of the positive class
brier_score = brier_score_loss(y_true, y_prob)

print('Brier score:', brier_score)
```

Conclusion

In conclusion, probability calibration is an essential step in the development of accurate and reliable machine learning models for classification. By using one of the methods discussed in this blog, you can ensure that your model's predicted probabilities are well-calibrated and that its predictions are accurate and trustworthy.

Binary Classification | Probability Calibration | Machine Learning

More from the list: "ML"

Curated by Himanshu Birla

Kyosuke... in Towards Dat...

Probability Calibration for Imbalanced Dataset

8 min read · Oct 20, 2019

Mattia Ci... in Analytics Vi...

How Probability Calibration Works

6 min read · May 28, 2020

Jason Yo...

Why Calibrat the Series on

7 min read · Oc

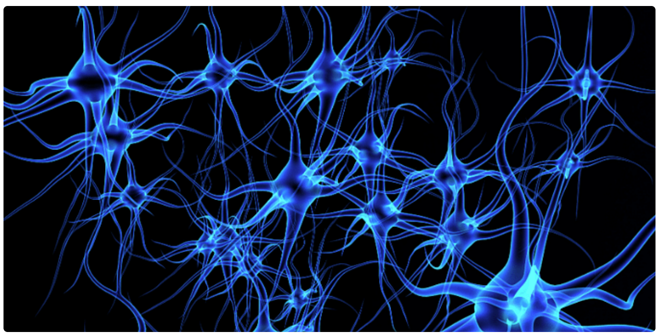
View list

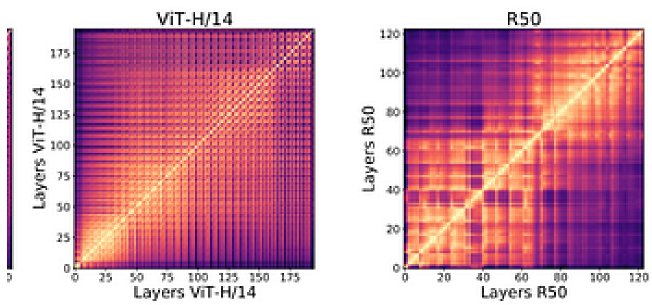
Written by Charlie Li

5 Followers

Following

Recommended from Medium





Vijay Reddiar

Multivariate time series forecasting and analysis of the US...

In the final part of this series we will compare the results of our modeling between classica...

17 min read · May 10

11

Ilias Papastratis

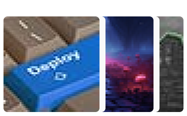
Comparison of Convolutional Neural Networks and Vision...

Introduction

19 min read · Sep 30

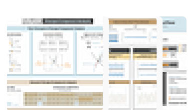
11

Lists



Predictive Modeling w/ Python

20 stories · 473 saves



Practical Guides to Machine Learning

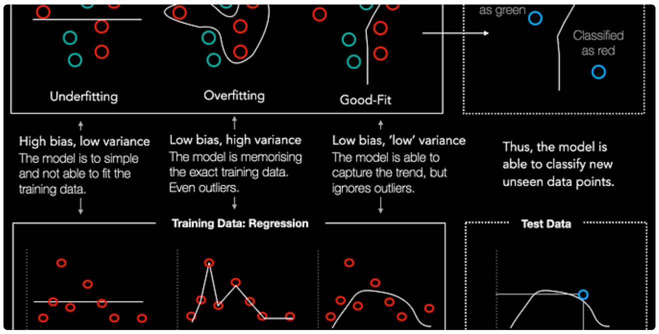
10 stories · 544 saves

Natural Language Processing

689 stories · 304 saves

The New Chatbots: ChatGPT, Bard, and Beyond

13 stories · 137 saves



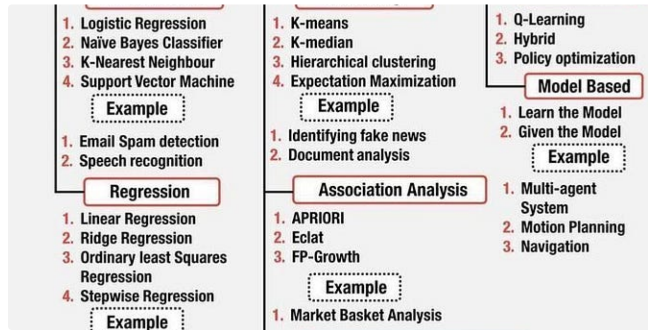
Frederik vl in Advanced Deep Learning

Understanding Bias and Variance in Machine Learning

The terms bias and variance describe how well the model fits the actual unknown data...

3 min read · Sep 15

106



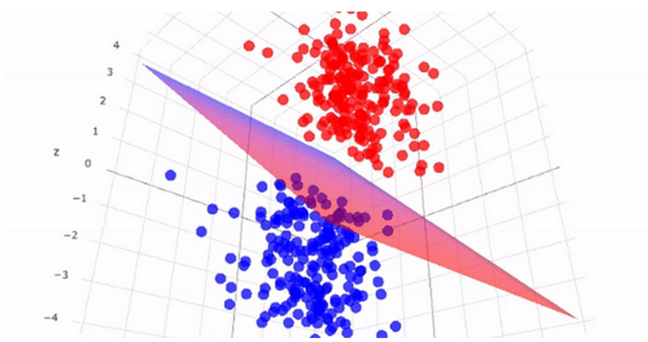
Mohsen Nabil

Machine Learning Algorithms ..

Hello ... I want to share some insights about the different types of Machine Learning...

2 min read · Jul 16

64



Tasmay Pankaj Tibre... in Low Code for Data Scie...

Support Vector Machines (SVM): An Intuitive Explanation

Everything you always wanted to know about this powerful supervised ML algorithm

Abel K Widodo

Exploring Heuristics Analytics with Python

In the era of big data and complex problem-solving, businesses and researchers are...