```
num = (O_g-np_g)**2
denom = np_g * (1-mpv)
ratio = num/denom
```

Then we compute the HL-statistics as

$$\chi^2_{HL} = \sum_{g=1}^{G} \frac{(O_g - n_g\hat{\pi}_g)^2}{n_g\hat{\pi}_g(1-\hat{\pi}_g)} = \frac{(0-0.03)^2}{0.03*(1-0.015)} + \frac{(1-0.07)^2}{0.07*(1-0.035)} + \frac{(1-0.12)^2}{0.12*(1-0.06)} = 19.6996$$

The p-value is lower than 0.0001 using the χ^2 distribution with 2 degrees of freedom. Hence, we reject the hypothesis of overall calibration for this significance level.

```
HL = np.sum(ratio).round(4)
pvalue = 1-chi2.cdf(HL,n_bins-1).round(4)

print('HL statistic:', HL)
print('p-value:', pvalue.round(decimals=4))
```

```
HL statistic: 19.6996
p-value: 0.0001
```

8.6 Metrics for Stability

Stability of a model means that the model and its parameters should not change over time. Hence, we want to test for model and parameter stability. This can be done if we have a parametric model, like a Logistic Regression. We create dummy variables for the two time periods for which we would like to test stability, include the interaction of the dummy variables with the features in the model specification, estimate the model and test whether the coefficients are different from zero. Formally, let a PD model with one variable be

$$PD_{it} = P(D_{it} = 1|x_{it-1}) = F(\beta_0 + \beta_1 \cdot x_{it-1})$$

where $F()$ is the link function. We then define a dummy variable

$$c_{t*} = \begin{cases} 1 & \text{if } t > t* \\ 0 & \text{otherwise} \end{cases}$$

and include the dummy into the regression, which becomes

$$PD_{it} = F(\beta_0 + \beta_1 \cdot x_{it-1} + \beta_2 \cdot c_{t*} + \beta_3 \cdot x_{it-1} \cdot c_{t*})$$

As a result, we have two regression equations

$$PD_{it} = F(\beta_0 + \beta_1 \cdot x_{it-1})$$

if $c_{t*} = 0$, i.e., $t \leq t*$ and

$$PD_{it} = F(\beta_0 + \beta_1 \cdot x_{it-1} + \beta_2 \cdot c_{t*} + \beta_3 \cdot x_{it-1} \cdot c_{t*})$$

if $c_{t*} = 1$, i.e., $t > t*$.

Hence, by testing whether β_2 and β_3 are different from zero, we actually test whether the coefficients of the model have changed over the two time periods.

In the next section, we will show the implementation during the case study with real data.

8.7 Function `validation`

In the `dcr` module, we have included the function `validation`, which provides a collection of validation techniques that we will apply throughout this book. The function provides an output of four panels:

- Upper left panel: summary table of validation metrics;
- Upper right panel: real-fit diagram by time, i.e., a comparison of the average observed and fitted outcome variable over time;
- Lower left panel: histogram of the fitted values;
- Lower right panel: real fit diagram by deciles of the fitted values.

This summary includes the visuals and metrics that we find most useful in validating models for default, payoff, loss rates given default and exposures.

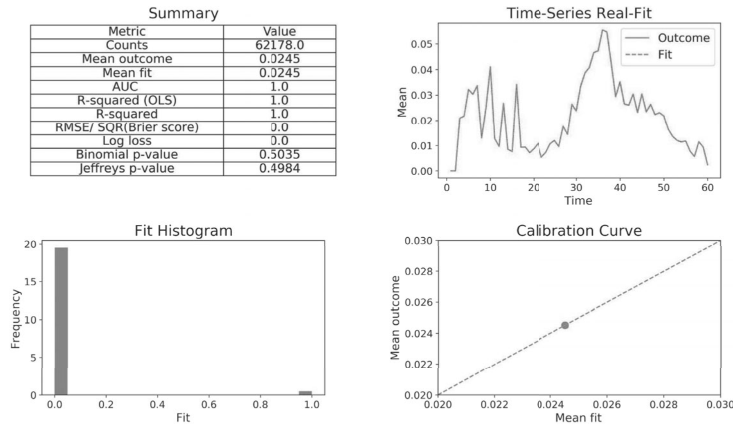
The function `validation()` has three input arguments:

- Outcome variable (may be binary or metric);
- Fitted variable (may be binary or metric);
- Time variable.

All variables may be `numpy` arrays or `pandas` dataframes. In a perfect PD model the fitted values are equal to the observed outcomes and the validation function provides the following output:

```
default_time = data.default_time.values
time = data.time.values

validation(default_time, default_time, time)
```



Summary

Metric	Value
Counts	62178.0
Mean outcome	0.0245
Mean fit	0.0245
AUC	1.0
R-squared (OLS)	1.0
R-squared	1.0
RMSE/ SQR(Brier score)	0.0
Log loss	0.0
Binomial p-value	0.5035
Jeffreys p-value	0.4984

In a perfect model, mean outcomes and mean fitted values match each other and performance measures are one, distance measures are zero and p-values are high as the hypothesis can not be rejected.

The summary table (upper left panel) reports observation counts, means for outcomes and fitted values, discrimination and calibration measures. Calibration measures are separated into R-squared, distances and p-values. The following metrics are included:

- Counts: number of observations;
- Mean outcome: average of outcome variable;
- Mean fit: average of fit variable if this value is close to the mean outcome then this is a good indication for calibration;
- AUC: area under the ROC, values are between zero (low fit) and one (high fit);
- OLS R-squared: this is the R-squared of an OLS regression of the outcome on the fit variable. The number is equal to the square of the Pearson correlation coefficient. Values are between zero (low fit) and one (high fit);
- `scikit-learn` R-squared: coefficient of determination as described above. Values are between minus infinity (low fit) and plus one (high fit);
- RMSE/SQR(Brier score): square root of the mean of the differences between outcome and fit variable;
- Log-loss: negative log-likelihood. Values are positive. The greater the value, the lower the fit;
- P-values of the Binomial test. The null hypothesis is that the PD is lower than or equal to the average predicted PD for the entire sample. Values are positive. The greater the value, the lower the fit;

- P-values of the Jeffrey's prior test whether the PD is lower than or equal to the average predicted PD for the entire sample. Values are positive. The greater the value, the lower the fit.

The real-fit diagram by time (upper right panel) shows that the default rate and mean fitted values match perfectly as they fully overlap.

The histogram (lower left panel) shows that the fitted values are either zero (higher number of observations) or one (lower number of observations).

The real-fit diagram by deciles of the fit variable (lower right panel) shows that the default outcomes and deciles of fitted values match perfectly.

8.8 Other Outcomes

Generally speaking, many validation metrics can be applied for binary outcomes (default and payoff events) and metric outcomes (LGD and EAD conversion measures). For example, Brier score and R-squared measures work for both binary and continuous outcomes.

However, some measures need to be applied with care. For example, in our validation function we are unable to compute for metric outcomes:

- Log loss;
- AUROC;
- Binomial p-value;
- Jeffrey's p-value.

For these measures, we compute the mean outcome ($\overline{outcome}$) and apply a categorization approach: the outcome variable is categorized into low (if outcome is below the mean: 0) and high (if outcome is above mean: 1):

$$\text{outcomeD} = \begin{cases} 1 & outcome \geq \overline{outcome} \\ 0 & outcome < \overline{outcome} \end{cases}$$

8.9 Validation Study

8.9.1 Data Preparation and Feature Engineering

We now analyze the mortgage data. We first read the data, and generate features using the function `dataprep`from module `dcr`. We assign the name _ to output datasets that we will not use here.

```
from dcr import *
data = data.query('time <=40').copy()

_, data_train, data_test, X_train_scaled, X_test_scaled, y_train, y_test = dataprep(data,
    depvar='default_time', splitvar='time', threshold=26)
```

The function `dataprep` creates the `numpy` arrays `X_train_scaled` and `X_test_scaled` for periods before and after period 26 (up to period 40). 11 features are included, scaled and ordered as follows: