
H2O.ai

H2O Target Mean Encoding

```
h2o.target_encode_apply(data, x, y, target_encode_map, holdout_type,
                        fold_column = NULL, blended_avg = TRUE,
                        noise_level = NULL, seed = -1)
```

```
def mean_target_encoding(data, x, y, fold_column):
    grouped_data = data[[x, fold_column, y]].group_by([x, fold_column])
    grouped_data.sum(na = "ignore").count(na = "ignore")
    df = grouped_data.get_frame().as_data_frame()
    df_list = []
    nfold = int(data[fold_column].max()) + 1
    for j in range(0, nfold):
        te_x = "te_{}".format(x)
        sum_y = "sum_{}".format(y)
        oof = df.loc[df[fold_column] != j, [x, sum_y, "nrow"]]
        stats = oof.groupby([x]).sum()
        stats[x] = stats.index
        stats[fold_column] = j
        stats[te_x] = stats[sum_y] / stats["nrow"]
        df_list.append(stats[[x, fold_column, te_x]])
    return h2o.H2OFrame(pd.concat(df_list))
```

Target Mean Encoding

Pay 1	Default Payment
Up To Date	0
Up To Date	0
Up To Date	0
Missed 1 Mo	1
Missed 1 Mo	0
Missed 1 Mo	0
Missed 5 Mo	1

H2O Target Mean Encoding

```
h2o.target_encode_apply(data, x, y, target_encode_map, holdout_type,
                        fold_column = NULL, blended_avg = TRUE,
                        noise_level = NULL, seed = -1)
```

```python
def mean_target_encoding(data, x, y, fold_column):
    grouped_data = data[[x, fold_column, y]].group_by([x, fold_column])
    grouped_data.sum(na = "ignore").count(na = "ignore")
    df = grouped_data.get_frame().as_data_frame()
    df_list = []
    nfold = int(data[fold_column].max()) + 1
    for j in range(0, nfold):
        te_x = "te_{}".format(x)
        sum_y = "sum_{}".format(y)
        oof = df.loc[df[fold_column] != j, [x, sum_y, "nrow"]]
        stats = oof.groupby([x]).sum()
        stats[x] = stats.index
        stats[fold_column] = j
        stats[te_x] = stats[sum_y] / stats["nrow"]
        df_list.append(stats[[x, fold_column, te_x]])
    return h2o.H2OFrame(pd.concat(df_list))
```