数据预处理

首先使用 `df.info()` 查看数据集的信息。

- 认为 id 一般来说不包含有意义的数据，所以先删除项 `id` 和 `id_str`。而有两个域 `utc_offset` 和 `time_zone` 对所有数据均为 `null`，应该直接删除。

```
d.drop(["id", "id_str", "utc_offset", "time_zone"], axis=1,
inplace=True)
```

- `entities` 域包含的数据格式较为复杂，难以解析。另外，有几个包含字符串，和 URL 的域，解析它们需要较为复杂的技术，可以先不处理，若之后效果需要提升，再考虑引入这些信息。

在做这些处理之后，所有数据都是 `non-null` 的。考虑处理那些非整形的数据。

- 有两个名字都是 `created_at` 的重名域，但它们是不同的数据，将第一个域重命名为 `created_at0`。第二个域的形式被判断为 `object`，将其转换为 `datetime`。

```
old_columns = list(d.columns[1:])
d.columns = ["created_at0"] + old_columns
d["created_at"] = pd.to_datetime(d["created_at"],
infer_datetime_format=True)
```

- 将标签信息单独提取为整形数据。将颜色值解析为整形数据。

```
df_label = pd.get_dummies(df["label"]).iloc[:, 0]
df.drop(["label"], axis=1, inplace=True)
```

```
d_rgb = d[name].apply(col2rgb)
d_rgb.columns = [name+"_r", name+"_g", name+"_b"]
d = pd.concat([d.drop([name], axis=1), d_rgb], axis=1)
```

- 最后对剩下的两类 `lang` 和 `translator_type` 直接使用 `pd.get_dummies()` 进行 one-hot 编码即可。注意需要补全的数据缺失了部分语言，需要保证训练数据和需要补全的数据有相同维度。另外 `test.json` 中同时存在 `en-GB` 和 `en-gb`，应该指同一种语言，因此将字符串转为小写。

```
def dummy2(a: pd.DataFrame, b: pd.DataFrame):
    N = len(a)
    concat_dummy = pd.get_dummies(pd.concat([a, b]))
    return concat_dummy[:N], concat_dummy[N:]
```

模型训练

数据集的拆分使用 `sklearn` 中的 `train_test_split` 即可。

线性模型