CS 559: Machine Learning: Fundamentals and Applications
Spring 2025 Project: Company Bankruptcy Prediction

1 Objective

The performance of logistic regression applied to the **original train data** is presented in Table 1. While the prediction of the target is highly accurate, identifying the companies that will file for bankruptcy ($y = 1$) is very poor due to the imbalance of the data (see Figure 1).

data set	FF(FT)	TT(TF)	accuracy score
train	5570(39)	5(193)	0.025
test	983(7)	0(22)	0

Table 1: Logistic Regression Result Table without any modification. The second and third columns present the number of companies' true status (y) and predicted status ($\hat{h}$) as FF - ($y = 0, \hat{h} = 0$), FT - ($y = 0, \hat{h} = 1$), TT - ($y = 1, \hat{h} = 1$), and TF - ($y = 1, \hat{h} = 0$). The accuracy score in the fourth column is evaluated using Equation (1).

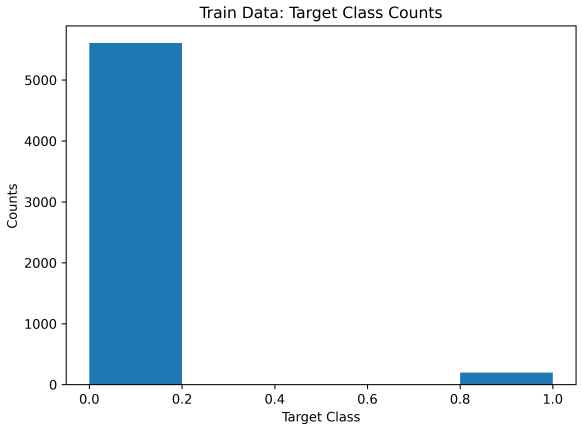

Figure 1: Train Data Target Distribution

The modeling objective is to identify the companies that will file for bankruptcy. Hereafter, the **accuracy score** (acc) is defined as how successfully the model identifies the company that files for bankruptcy as follows

$$acc = \frac{TT}{TF + TT}. \tag{1}$$

1.1 Class Competition

This project is a class competition, and teams will be ranked using the following metrics:

$$Rank = 0.3(acc_{train}) + 0.4(acc_{test}) + 0.3\left(\frac{50 - N_{features}}{50}\right)$$

where

- acc_{train} is the accuracy score of the train model from Section 3.3.
- acc_{test} is the accuracy score of generalization obtained from Section 4.

- $N_{features}$ is the number of features used in the model.

The rank will be converted to the top percentile in the competition and will be used for a portion of the individual's project grade (see Section 5.3).

2 Data Description

The train and test data sets have 5807 rows and 1012 rows, respectively. The total number of given features is 95. The project aims to train a model that identifies whether a company will file for bankruptcy (the target column is "Bankrupt?") or not. The test data set does not have the target column. Each team will submit the predicted classes for 1012 companies to the submission file.

3 Model

Since the data is highly imbalanced, logistic regression could not capture the crucial features for companies with $y = 1$. Therefore, the model needs to be trained differently to account for crucial features among companies with $y = 1$. The new proposed method is to create a classification model for companies in similar conditions.

3.1 Training Data Preparation

95 features are considered too many features. The number of features **must be reduced** to increase the model's efficiency. Each team will produce a new training data set with no more than 50 features. Be sure to drop the "Bankrupt?" and "Index" columns, but note that they will be used later. There are unlimited ways and numbers to extract, select, drop, and engineer the remaining features.

1. The new train data must not be multicollinear.
2. Each feature must be in its own Gaussian.

3.2 Company Characterization

Considering the data size, understanding every company's situation is quite difficult at a given time. However, knowing the general situation will greatly help model training. The easiest approach is finding the common characteristics between similar companies, and similar companies can easily be grouped by using unsupervised learning clustering techniques.

1. Cluster the training data from 3.1 into k-many subgroups where k can be up to twice the team size but not less than the team size. For example, if a team has three members, the cluster groups can be between 3 and 6. This allows each team member to create the prediction model for at least one subgroup. If you find that a subgroup has no bankrupted companies among the train data, then do not assign it to a team member, but instead assign it a constant prediction model ($\hat{h} = 0$). If this does not leave enough subgroups for each team member to be assigned at least one, then increment k by one, and try again until each team member is assigned at least one subgroup.
2. Report the number of companies and the proportion of bankrupted companies in each subgroup.
3. Identify unique or helpful characteristics in each subgroup. Use visualization techniques to present identified characteristics.
4. Keep the cluster IDs. They will be used later in Section 3.3.

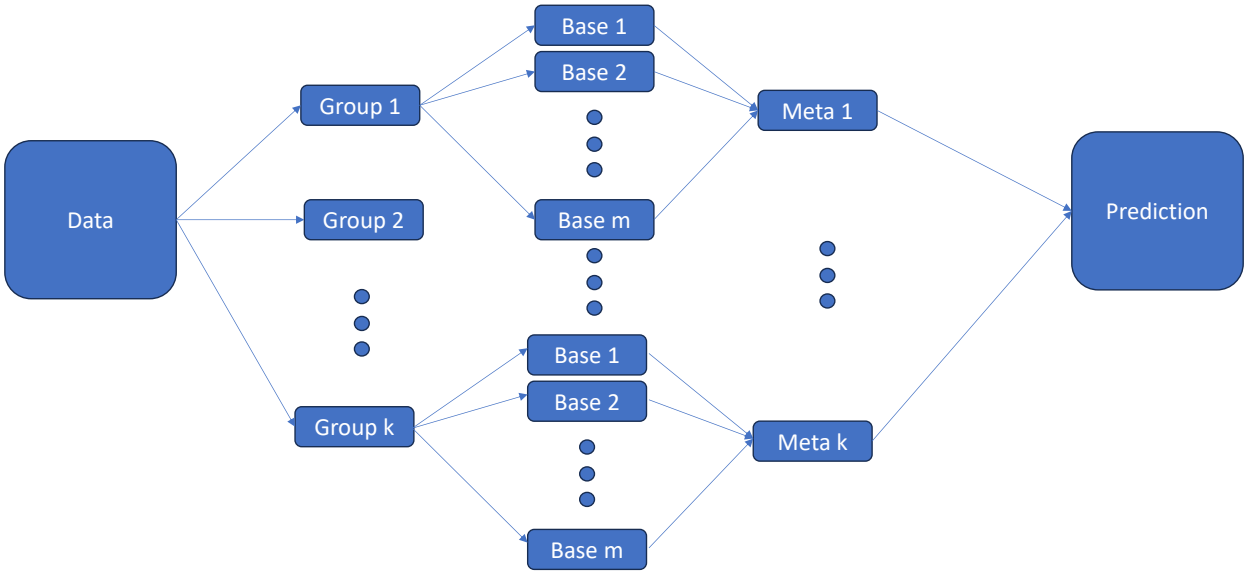

Figure 2: Stacking Method Training Model Diagram. The diagram shows the k-many subgroups, m-many base models in each subgroup, and k-many meta-models where each meta-model is a combined model of m-base models in each subgroup.

3.3 Bulding Training Models

It can be broken into two steps. The first step is to predict the group (the cluster-ID from Section 3.2) the company will likely belong to, and the second step is to classify whether the company will file for bankruptcy in each subgroup. Figure 2 displays the roadmap of the model.

1. Build a classification model that predicts a subgroup a company will likely belong to using any supervised learning algorithm. The prediction's accuracy should be high, and it is okay to overfit. Identify the features that play important roles in this prediction.
2. Build a stacking model (with cross-validation) that predicts whether a company in the subgroup will file for bankruptcy.
    a. The base models must be non-parametric. Have three or more.
    b. Each subgroup must use the same features for all base models. However, features can be different between subgroups.
    c. Each member must build a model for at least one subgroup. The member's name, the subgroup(s) worked on, and *acc* must be reported. Provide the confusion matrix. See Table 3 for the reporting format.

4 Generalization

Transform the test data set as the training set in Section 3.1. Then, predict each company's bankruptcy status in the test data. Paste the result to the submission file (Table 2). The instructor will evaluate the

accuracy score using the predicted class the team submitted.

Index	Bankrupt?
1	0
2	1
⋮	⋮
1012	0

Table 2: Submission File Example

5 Timeline, Submission, Grade Scheme

The submission of the project is due on April 29 by 11:59 PM.

5.1 Timeline

The ideal timeline of the project is broken down as follows.

1. Train Model: The train data set will be released on Tuesday, 4/1, so the training data preparation and building of a training model can be started.
2. Generalization: **The test data will be released on the second week of Project Weeks (4/15)**. The generalization should be one-time work and should not take a long time.
3. The video presentation and files must be submitted by 4/29.

5.2 Submission

Here are the requirements for submission. Each team will submit a single compressed **zip** file. The submission will require the following files inside the zip file. Name the file as *GroupNumber_CourseSection.zip*.

1. Sections 3.1 and 3.2: A single team notebook file: Clean and summarize the work. Use Markdown for comments and explanations. Name the file as *GroupNumber_TrainingData*.
2. Section 3.3: A single notebook file for individual team members. Show the work and summarize the work. **Individual files must be included in the compressed file**. If any team member fails to submit, the member will be excluded from the grading. Name the file as *MemberName_Subgroup#*. If any member worked on multiple subgroups, submit the file for each.
3. Section 4: Submit a single team's notebook file with code and the submission file as shown in Table 2. Name the file as *GroupNumber_Generalization*.
4. Result Summarization: Summarize the results as a team in the docx file.
   - Section 3.2: Report the number of subgroups, the number of $y = 0$ and $y = 1$, and summarize characteristics, distributions, or properties of companies in each subgroup.
   - Section 3.3: Construct a table as shown in Table 3. In addition to that, construct a confusion table for base models and the meta-model for each subgroup.
   - Video Presentation: Record the video presentation to describe the workflow, models, and results. You can either include the video file in the zip file or provide the link in the summarization document.

   ** All reported results must be consistent with notebook files. Make sure to use **random.seed()** or **random_state=** whenever needed so the same results can be reproduced. Each notebook file must display the result. Any non-consistent results will be marked 0.
   - Name the file as *GroupNumber_CourseSection_Results*

Subgroup ID	Name of Student	Average accuracy score base models [TT(TF)]	accuracy score Meta model [TT(TF)]	$N_{features}$
1	John	0.93 [184(14)]	0.84 [167(31)]	34
2	Kim	0.80 [140(34)]	0.98 [170(4)]	6
3	Constant	1.00 [0(0)]	1.00 [0(0)]	0
⋮	⋮	⋮	⋮	⋮
k	Name	0.xx [$\cdots$]	0.xx [$\cdots$]	xx
team		0.xx [$\cdots$]	0.xx [$\cdots$]	xx

Table 3: Result Table Example: Column 1: The subgroup ID from Section 3.2, column 2: the name of the member who worked on the subgroup, or 'Constant' denoting a subgroup with no bankruptcies, column 3: the average of the training accuracy score (Equation 1) from base-models in the subgroup, column 4: the training accuracy score from the meta-model of the subgroup, column 5: the number of features the member used in the stacking model. As a team score, the last row gives average *acc* scores, sum TT(TF), and average $N_{features}$.

5.3 Grade Scheme

This is how the project will be graded. Although this is a team project, each member will be graded separately, as shown below

$$Score = 0.2(Rank) + 0.4(acc_{Table3}) + 0.4\left(\frac{50 - N_{features,Table3}}{50}\right) \tag{2}$$

where $Rank$ is the top percentile of the competition, acc_{Table3} is the accuracy score of the individual's train model in Section 3.3*, and $N_{features,Table3}$ is the number of features the individual used in the training model in Section 3.3. The accuracy scores and number of features reported in Table 3 will be used. The accuracy metrics in Equation 1 will be used for the individual's grade.

Note*: If a member trained the stacking model for more than one subgroup, the best-performed model in terms of accuracy will be used in the grading metric.

Note: The group performance ($Rank$) will weigh 20% of the project grade, and the remaining 80% of the grade will be evaluated from the individual's performance.

Note: This is a team project. If any member of a team is not responsive and does not return emails, texts, or calls, it is the team's responsibility to contact me. If this member was reported twice or more, the student will be out of the project and receive a zero. Please be responsive to team members.

整体语境

师是在给出一些更准确的规范解释，主要集中在以下三个方面：

1. 项目评分中 feature 数量的计算方式 Clarification
2. 关于 cluster 分析后如何处理子模型数据格式的灵活处理 Clarification
3. 关于两个 team notebook 文件内容分工 Clarification

---

1. 关于 N_features 的计算

_features 是排名公式里的一个因子，原说明中描述为“模型使用的特征数”。老师澄清：

- 不是所有子模型用特征数的普通平均值；
- 而是：加权平均（weighted average）；
- 权重 = 每个子群（subgroup）所占的训练数据比例；

你在 Table 3 最后一行（团队汇总）中填的 N_features 应该是：

_{\text{features, team}} = \sum_{i=1}^{k} \left(\frac{n_i}{n} \times N_{\text{features, subgroup}_i} \right)

---

2. 关于 clustering 之后的子模型处理流程：

原文提法让大家默认认为：“分好 cluster 的数据直接就被送去给子模型建模。”

师说其实你有更多灵活操作空间，而且这些操作可能让你建模更方便、表现更好：

推荐思路如下：

1. cluster analysis 阶段
   - 你做特征选择、归一化等 → 得到 cluster label；
   - 这一步你用自己的 transform 没问题；
2. cluster label 预测模型阶段（Section 3.3 第一步）
   - 你可以重新用原始数据来训练一个模型预测 cluster label；
   - 这样可以分析“哪些原始特征在预测 cluster 时作用大”，符合规范要求；
3. subgroup 模型阶段（Section 3.3 第二步）
   - 子组成员可以只拿某个 cluster 的原始数据，再独立做 preprocessing；
   - 这相当于：每个子组可以当自己是做一个新的模型任务，有独立的建模过程；

关键点：你只需要把原始数据的 row index 和它的 cluster label 对应起来保存，后面都可以自由重新开始。

---

3. 关于两份团队 notebook 的内容分工：

师说得很清楚：

们要交两份团队 notebook：

- GroupNumber_TrainingData.ipynb
- GroupNumber_Generalization.ipynb

常见误区：

家以为 generalization notebook 要重复很多 training 的代码（特征工程、模型加载、预测）

确做法（推荐）：

你的预处理 pipeline 和模型保存成对象（.pkl、.joblib 等），比如：

```
om sklearn.externals import joblib
oblib.dump(model, 'model.pkl')
oblib.dump(scaler, 'scaler.pkl')
```

然后在 generalization notebook 中：

```
odel = joblib.load('model.pkl')
caler = joblib.load('scaler.pkl')
```

这样你就只需要加载并应用，而不是重复训练。

这是一种非常好的实践（Best Practice）：强制你写出可复用、可模块化的训练过程，同时避免 notebook 内容重复太多。

总结：你现在该怎么做?

项目部分	推荐调整方式
able 3 最后一行	使用子组样本数量加权平均计算 N_features
luster label 模型	可以使用原始数据重新构造特征用于分类，别被限制住
ubgroup 模型	每个成员可以用原始数据做自己的 preprocessing 和模型建构
eneralization notebook	不要 copy-paste training notebook，用 joblib/pickle 保存模型与 transform，再加载

要我帮你写一个：