Зеленков Ю.А., Володарский Н.С. Классификация несбалансированных данных как проблема многокритериальной оптимизации.

Большинство задач классификации, решаемых на практике, имеют дело с несбалансированными наборами данных, когда представительство разных классов в обучающей выборке значительно отличается. При этом, с одной стороны, возникает проблема оценки качества построенной модели, в качестве метрик при бинарной классификации обычно используют геометрическое среднее $G_{mean} = \sqrt{TNR * TPR}$ и площадь $AUC = (TPR + TNR)/2$, где $TPR = TP/(FN + TP)$, $TNR = TP/(TN + FP)$ – доля верно классифицированных объектов положительного и отрицательного классов соответственно, переменные FN, TP, TN, FP соответствуют значениям в ячейках матрицы ошибок (таблица 1). С другой стороны, наличие дисбаланса классов ограничивает использование алгоритмов машинного обучения, поскольку при построении модели они оптимизируют функцию, также ориентированную на сбалансированные данные.

Таблица 1. Матрица ошибок

Действительный класс	**Предсказанный класс**	
	Отрицательный	Положительный
Отрицательный	*TN*	*FP*
Положительный	*FN*	*TP*

В данной работе предлагается рассматривать задачу классификации как проблему многокритериальной оптимизации и строить классификатор, одновременно минимизируя параметры $FPR = FP/(TN + FP)$ и $FNR = FN/(FN + TP)$, т.е. доли неверно классифицированных объектов отрицательного и положительного классов соответственно.

Рассмотрим задачу многокритериальной минимизации без ограничений с m независимыми переменными и n целями

$$\min_{\mathbf{x}} \mathbf{f}(\mathbf{x}),$$

где $\mathbf{x} = (x_1, x_2, \dots, x_m) \in X$ – вектор решений (независимых переменных), X – пространство параметров, $\mathbf{f}(\mathbf{x})^T = [f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_n(\mathbf{x})]$ – цели. Вектор $\mathbf{a} \in X$ доминирует вектор $\mathbf{b} \in X$ (обозначается как $\mathbf{a} \prec \mathbf{b}$) если выполняется условие $\forall i \in \{1, \dots, n\}: f_i(\mathbf{a}) \le f_i(\mathbf{b}) \wedge \exists j \in \{1, \dots, n\}: f_j(\mathbf{a}) < f_j(\mathbf{b})$. Множество решений X', для которого выполняется условие $\forall \mathbf{a}' \in X': \neg \mathbf{a} \prec \mathbf{a}' \wedge \|\mathbf{a} - \mathbf{a}'\| < \varepsilon \wedge \|f(\mathbf{a}) - f(\mathbf{a}')\| < \delta$, где $\|\dots\|$ - метрика расстояния и $\varepsilon > 0, \delta > 0$, называется локальным Парето-оптимальным множеством. X' является глобальным Парето-оптимальным множеством, если $\forall \mathbf{a}' \in X': \neg \mathbf{a} \prec \mathbf{a}'$.

В рассматриваемом здесь случае $n = 2$ и $\mathbf{f}(\mathbf{x})^T = [FNR, FPR]$. Предлагаемый алгоритм состоит из двух шагов. Первый шаг - обучение пула гетерогенных классификаторов, второй шаг – отбор классификаторов и построение ансамбля, оптимизирующего комбинацию $[FNR, FPR]$.

На первом шаге мы использовали «классические» модели (Gradient Boosting, AdaBoost, Bagging, Random Forest, Extra Trees и логистическая регрессия [1]), калиброванный AdaBoost [2], а также методы динамического отбора классификаторов и их ансамблей (APriori, MCB, OLA, DESP, KNORAE, KNORAU и METADES [3]) как на необработанных данных, так и в сочетании с различными методами over (SMOTE, ADASYN) и under сэмплинга (Random Under Sampling и Tomek Links) [4], всего 70 различных алгоритмов. Результаты кросс-валидации ($k = 3$) этих моделей на наборе данных [5] о банкротствах российских компаний

(2457 наблюдений, из которых 456 представляют положительный класс), представлены на рис. 1. Точки красного цвета соответствуют моделям, принадлежащим Парето-оптимальному множеству, синие точки – неоптимальным решениям.

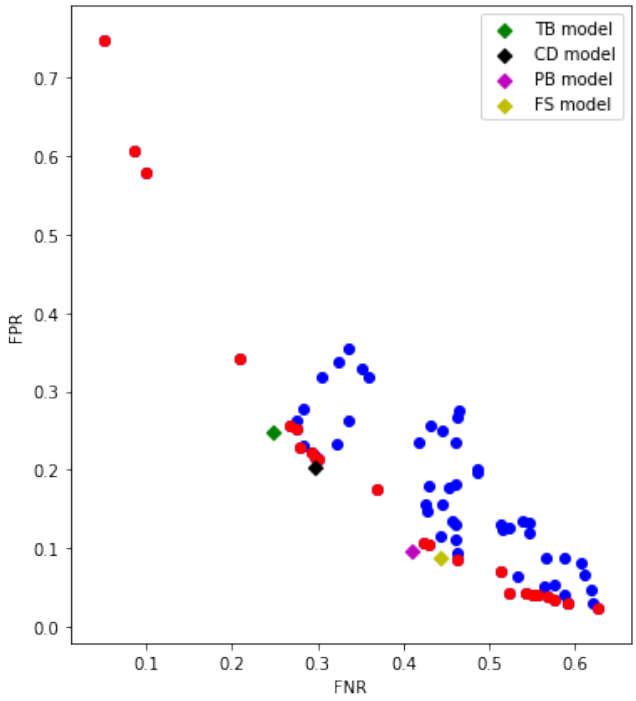

Рис. 1 Результаты кросс-валидации ($k = 3$) моделей в координатах *FNR*, *FPR*.

Для реализации второго шага были рассмотрены следующие варианты:

- Объединение всех моделей в пуле (FS model);
- Объединение моделей, чьи результаты принадлежат Парето-оптимальному множеству (PB model);
- Объединение моделей, чьи результаты принадлежат Парето-оптимальному множеству, затем исключение моделей, результаты которых незначительно отличаются от других, на основе алгоритма crowding distance [6] (CD model). Это позволяет увеличить разнообразие моделей в ансамбле.
- Объединение моделей, чьи результаты принадлежат Парето-оптимальному множеству, и для которых выполняется условие $FNR \leq t \wedge FPR \leq t$, где t – пороговое значение (TB model). Это равносильно «вырезанию» центральной части Парето-оптимального множества, тем самым отбираются модели, обеспечивающие более сбалансированную комбинацию $[FNR, FPR]$.

Таблица 2. Результаты тестирования предложенных моделей (кросс-валидация, $k = 3$)

Модель	Размер ансамбля	G_{mean}	*AUC*	*FPR*	*FNR*
Best estimator	1	0.746	0.747	0.228	0.279
Averaging of all models (FS)	70	0.712	0.734	0.089	0.443
Threshold based (TB) model	8	0.752	0.752	0.247	0.248
Crowding distance (CD) based model	12	0.749	0.751	0.203	0.296
Pareto based (PB) model	23	0.730	0.747	0.096	0.410

Результаты кросс-валидации предложенных моделей ($k = 3$) представлены на рис. 1 и в таблице 2. Как следует из рис. 1, все варианты, за исключением FS модели, продуцируют ансамбли, результаты которых

сдвигаются относительно Парето-оптимального множества базовых классификаторов к началу координат, что свидетельствует об улучшении их предиктивной способности.

В таблице 2 представлено количество базовых моделей, объединенных в ансамбль согласно каждому из рассмотренных методов, и соответствующие значения метрик качества, а также наилучшие значения этих метрик, достигнутые лучшим классификатором из исходного пула. В данном случае это алгоритм Random Forest (100 базовых классификаторов), обученный на тестовой выборке, прошедшей препроцессинг с помощью метода Random Under Sampling (RUS).

Как следует из представленных данных, лучшие значения G_{mean} и AUC демонстрирует модель, объединяющая классификаторы, принадлежащие центральной части Парето-оптимального множества (TB model) с пороговым значением в данном случае $t = 0.4$. При этом достигаются практически равные значения *FPR* и *FNR*, а также наименьший размер ансамбля.

Литература

1. Pedregosa F. et al. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12: 2825-2830.
2. Nikolaou, N. et al. (2016). Cost-sensitive boosting algorithms: Do we really need them*? Machine Learning*, 104(2-3): 359-384.
3. Cruz, R. M., Sabourin, R., & Cavalcanti, G. D. (2018). Dynamic classifier selection: Recent advances and perspectives. *Information Fusion*, 41: 195-216.
4. Lemaitre, G., Nogueira, F., & Aridas, C.K. (2017). Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning. *Journal of Machine Learning Research*, 18(17): 1-5.
5. Zelenkov, Y., Fedorova, E., & Chekrizov, D. (2017). Two-step classification method based on genetic algorithm for bankruptcy forecasting. *Expert Systems with Applications*, 88: 393-401.
6. Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation*, 6(2): 182-197.