Bài 2.22 A Comparative Analysis of Machine Learning Algorithms

Phân tích so sánh các thuật toán học máy trong phân tích dự đoán: Ứng dụng trên dữ liệu kinh tế Việt Nam

Nguyễn Xuân Trường[1]

[1]Khoa Hệ thống thông tin, Trường Đại học Công nghệ Thông tin, Thành phố Hồ Chí Minh, Việt Nam, truongnx.18@grad.uit.edu.vn

A. TÓM TẮT ĐỌC HIỂU BÀI BÁO

Bài báo trình bày về việc so sánh toàn diện các thuật toán học máy trong việc phân tích dự báo ở lĩnh vực chăm sóc sức khỏe. Tác giả sử dụng các thuật toán học máy để đánh giá các công việc dự báo bao gồm: Dự đoán tái nhập viện, chuẩn đoán bệnh, đánh giá rủi ro của bệnh nhân, phát hiện gian lận và phát hiện bệnh sớm. Bài nghiên cứu này sử dụng các thuật toán như: Cây Quyết Định (Decision Trees), Rừng Ngẫu Nhiên (Randome Forests), SVM (Support Vector Machines), Mạng Neural (Neural Networks), k-NN (k-Nearst Neighbors), và Gradient Boosting [1], mục đích của bài toán đặt ra là phân loại bệnh cho nên tác giả nghiên cứu đã sử dụng các độ đo như: Độ chính xác (Accuracy), Độ chuẩn xác (Precision), Độ nhạy (recall), Điểm F1 (F1-score) và ROC-AUC (Receiver Operating Characteristic - Area Under Curve).

Từ khóa: Healthcare, Predictive Analytics, Machine Learning, Comparative Analysis, Algorithm Performance, Gradient Boosting, Patient Care, Data Science, Healthcare Industry.

1. Giới Thiệu

1.1 Tổng quan và bối cảnh

Trước khi bắt đầu đi sâu vào trong nghiên cứu, tác giả bài báo đã chỉ ra những sự thay đổi đáng kể của lĩnh vực chăm sóc sức khỏe trong những năm gần đây. Cụ thể, ngành chăm sóc sức khỏe đã trải qua những biến đổi nhờ vào việc dữ liệu bệnh nhân ngày càng lớn và nhu cầu đưa quyết định dựa trên dữ liệu ngày càng tăng. Sự chuyển đổi này được cho là do sự xuất hiện của phân tích dự đoán, một công cụ sử dụng các kỹ thuật thống kê tiên tiến và học máy để rút ra những thông tin quý giá từ dữ liệu chăm sóc sức khỏe. Phân tích dự đoán trong lĩnh vực chăm sóc sức khỏe đã trở thành một thành phần quan trọng trong việc cải thiện kết quả cho bệnh nhân, tăng cường hiệu quả hoạt động và giảm chi phí.

Trước các yêu cầu và chi phí chăm sóc sức khỏe ngày càng gia tăng, các nhà cung cấp dịch vụ y tế, nhà thanh toán và nhà hoạch định chính sách đang hướng đến phân tích dự đoán như một công cụ để giải quyết các công việc phức tạp. Việc áp dụng phân tích dự đoán trong chăm sóc sức khỏe không chỉ hứa hẹn mang lại chăm sóc bệnh nhân hiệu quả hơn mà còn tiềm năng biến đổi toàn bộ hệ sinh thái chăm sóc sức khỏe.

1.2. Mục tiêu nghiên cứu, câu hỏi và giả thuyết

Bài báo nghiên cứu về các thuật toán học máy dùng cho phân tích dự đoán được sử dụng trong lĩnh vực chăm sóc sức khỏe. Tiến hành đánh giá hiệu suất của các thuật toán dựa theo tiêu chí và độ đo phù hợp. Xác định các điểm mạnh và điểm ý của các thuật toán khác nhau trong ứng dụng chăm sóc sức khỏe. Đưa đến các đề xuất cho việc lựa các thuật toán học máy cụ thể trong từng công việc ở lĩnh vực chăm sóc sức khỏe. Cuối cùng là những đóng góp của bài báo để giải quyết sự khác biệt giữa các nghiên cứu trước đây.

Câu hỏi nghiên cứu: Bài báo đưa ra câu hỏi thuật toán học máy nào là hiệu quả nhất trong phân tính dự đoán trong lĩnh vực chăm sóc sức khỏe, và làm thế nào để so sánh về mặt Độ chính xác (Accuracy), Độ chuẩn xác (Precision), Độ nhạy (recall), Điểm F1 (F1-score) khi ứng dụng vào các công việc dự đoán khác nhau.

Với câu hỏi trên, tác giả đã cho ra 2 giả thuyết là: Thứ nhất, không có sự khác biệt giữa các thuật toán trong các công việc dự báo. Thứ 2 là có sự khác biệt giữa thuật toán khi ứng dụng vào trong phân tích dự đoán được thể hiện về mặt Độ chính xác (Accuracy), Độ chuẩn xác (Precision), Độ nhạy (recall), Điểm F1 (F1-score).

2. Tổng Quan Nghiên Cứu

Trong phần này tác giả tổng hợp và đánh giá 12 kết quả nghiên cứu từ 2010 đến 2021 về phân tích đoán trong lĩnh vực chăm sóc khỏe. Với mỗi bài báo tác giả đã chỉ ra những biến quan trọng (key variables) và những phát hiện chính của từng bài báo (key findings). Ví dụ bài báo năm 2021 của tác giả Smith et at., "Predicitive Modeling in ICU", tác giả bài báo đã chỉ ra biến quan trọng là: Thông số sinh lý của bệnh nhân (Patient vitals), các bệnh đồng bệnh kèm (comorbidities), kết quả xét nghiệm hóa sinh (lab results) và phát hiện chính là đưa ra được mô hình đoán ICU theo thuật toán Rừng ngẫu nhiên (Random Forest) đạt được độ chính xác cao trong kết quả điều trị bệnh nhân.

Sau khi đánh giá các bài báo, tác giả đã thấy những bài báo trước đó có một khoảng trống lớn vẫn chưa được giải quyết. Tác giả chỉ ra các nghiên cứu trước đó tập trung vào các khía cạnh cụ thể như dự đoán nguy cơ tái nhập viện, dự báo dịch bệnh, hoặc tuân thủ điều trị thuốc. Mà những nghiên cứu này lại sử dụng nhiều thuật toán học máy và nguồn dữ liệu khác nhau, chính vì thế làm cho việc xác định được thuật toán nào hoạt động tốt nhất trở nên khó khăn.

Dựa trên kết quả đánh giá bài báo, tác giả đã đề xuất một mô hình đánh giá thuật toán thống nhất. Từ đó cho ra một kết quả phân tích so sánh đầy đủ để hỗ trợ người hoạt động trong lĩnh vực chăm sóc sức khỏe và nhà nghiên cứu dữ liệu đưa ra quyết định chính xác hơn dựa theo các công việc phân tích cụ thể cho từng thuật toán.

3. Phương Pháp Nghiên Cứu

Tác giả đã phác thảo ra mô hình nghiên cứu, nguồn dữ liệu, phương thức thu thập dữ liệu và công cụ phân tích dữ liệu.

Đầu tiên, tác giả xác định mô hình nghiên cứu là Phân tích so sánh (Comparative Analysis), sau đó xác định nguồn dữ liệu trong lĩnh vực chăm sóc sức khỏe (Healthcare).

3.1 Phương thức thu thập dữ liệu.

1. Truy xuất dữ liệu: Thu thập các bộ dữ liệu chăm sóc sức khỏe có sẵn công khai với các nhiệm vụ dự đoán đa dạng và các đặc trưng khác nhau.
2. Tiền xử lý dữ liệu: Làm sạch, biến đổi và tiền xử lý dữ liệu để đảm bảo tính nhất quán và chất lượng.
3. Triển khai thuật toán: Thực hiện một loạt các thuật toán học máy:
    a. Cây Quyết Định (Decision Trees),
    b. Rừng Ngẫu Nhiên (Randome Forests),
    c. SVM (Support Vector Machines),
    d. Mạng Neural (Neural Networks),
    e. k-NN (k-Nearst Neighbors),
    f. Gradient Boosting.
4. Kiểm định chéo k-fold: Áp dụng kiểm định chéo k-fold để đánh giá tính tổng quát của thuật toán và giảm thiểu hiện tượng quá khớp (over fitting).
5. Các chỉ số hiệu suất: Tính toán độ chính xác, độ chuẩn xác, độ nhạy, điểm F1 và ROC-AUC cho mỗi thuật toán.
6. Phân tích so sánh: So sánh hiệu suất của các thuật toán trên các nhiệm vụ dự đoán khác nhau trong lĩnh vực chăm sóc sức khỏe.

3.2 Công cụ phân tích dữ liệu

1. Lập trình Python: Sử dụng ngôn ngữ lập trình Python để tiền xử lý và phân tích dữ liệu.
2. Scikit-Learn: Sử dụng thư viện Scikit-Learn để triển khai và đánh giá các thuật toán học máy.
3. Kiểm định thống kê: Thực hiện các kiểm định thống kê (ví dụ: t-test, ANOVA) để đánh giá sự khác biệt có ý nghĩa về hiệu suất của các thuật toán.
4. Trực quan hóa dữ liệu: Tạo các biểu đồ và đồ thị để minh họa hiệu suất của các thuật toán.

3.3 Nguồn dữ liệu

Tác giả sử dụng các nguồn dữ liệu công khai nhưng uy tín để đảm bảo chất lượng và độ tin cậy dữ liệu. Những bộ dữ liệu đó bao quát các khía cạnh công việc dự đoán như dự báo khả năng nhập viện, đánh giá rủi ro bệnh,…

Bằng việc tuân theo phương pháp luận nghiên cứu một cách nghiêm ngặt, tác giả đã tiến hành phân tích toàn diện các thuật toán học máy trong ngữ cảnh phân tích dự báo ở lĩnh vực chăm sóc sức khỏe, và giải quyết các nghiên cứu đã chỉ ra ở phần giới thiệu.

4. Kết Quả

Sau khi thử nghiệm các thuật toán học máy cho 5 ngữ cảnh khác nhau trong lĩnh vực chăm sóc sức khỏe theo phương pháp triển khai ở trên, tác giả nhận thấy thuật toán Gradient Boosting cho ra kết quả vượt trội so với các thuật toán còn lại dựa theo các độ đo cho bài toán phân loại.

B. CÀI ĐẶT MINH HỌA TRÊN BỘ DỮ LIỆU VỀ KINH TẾ

1. Truy xuất dữ liệu

Dữ liệu được sử dụng trong nghiên cứu này là dữ liệu về 42.639 khách hàng của một ngân hàng Việt Nam được trích dẫn từ Kaggle [4] . Dữ liệu bao gồm 21 biến, trong đó có 20 biến đầu vào và 1 biến mục tiêu là liệu khách hàng có đăng ký gửi tiền có kỳ hạn hay không ('term_deposit').

Bảng 1: Ký hiệu và định nghĩa biến theo các đặc điểm dữ liệu

Biến đầu vào	**Định nghĩa biến**
Thông tin nhân khẩu học	
'age'	Tuổi
'job'	Loại công việc
'marital'	Tình trạng hôn nhân
'education'	Trình độ học vấn
Thông tin tài chính	
'default'	Nợ xấu
'housing'	Vay mua nhà
'loan'	Vay cá nhân
Thông tin liên lạc	
'contact'	Loại hình liên lạc
'month'	Tháng liên lạc gần nhất
'dayofweek'	Ngày liên lạc gần nhất
'duration'	Thời gian liên lạc
Thông tin chiến dịch tiếp thị	
'campaign'	Số lượng liên hệ được thực hiện trong chiến dịch này và cho khách hàng này (số, bao gồm liên hệ cuối cùng)
'pdays'	Số ngày kể từ lần liên lạc cuối cùng của chiến dịch trước
'previous'	Số lần liên lạc trước chiến dịch này
'poutcome'	Kết quả chiến dịch trước
Thông tin kinh tế xã hội	
'emp.var.rate'	Tỷ lệ biến động việc làm
'cons.price.idx'	Chỉ số giá tiêu dùng
'cons.conf.idx'	Chỉ số niềm tin người tiêu dùng
'euribor3m'	Lãi suất euribor 3 tháng
'nr.employed’	Số lượng nhân viên

Các biến phân loại như: 'job', 'marital', 'education', 'default', 'housing', và 'loan' sẽ được mã hóa để phù hợp với yêu cầu của các thuật toán học máy.

2. Tiền xử lý dữ liệu

Dữ liệu được xử lý để đảm bảo tính nhất quán và loại bỏ các giá trị ngoại lai. Các bước tiền xử lý bao gồm:

- **Xử lý giá trị thiếu:** Các giá trị thiếu trong các biến số *('age', 'emp.var.rate', 'cons.price.idx', 'cons.conf.idx', 'euribor3m', 'nr.employed')* được thay thế bằng giá trị trung bình hoặc giá trị nội suy của các quan sát xung quanh. Các giá trị thiếu trong các biến phân loại *('job', 'marital', 'education', 'default', 'housing', 'loan')* được thay thế bằng giá trị 'unknown'.
- **Mã hóa dữ liệu phân loại:** Các biến phân loại *('job', 'marital', 'education', 'default', 'housing', 'loan')* được mã hóa bằng phương pháp One-Hot Encoding [5][6] để chuyển đổi chúng thành dạng số, phù hợp với yêu cầu của các thuật toán học máy.
- **Chia dữ liệu:** Dữ liệu được chia thành hai tập: tập huấn luyện (80%) và tập kiểm tra (20%). Tập huấn luyện được sử dụng để xây dựng mô hình, trong khi tập kiểm tra được sử dụng để đánh giá hiệu suất của mô hình.

3. Triển khai thuật toán

Dựa trên kết quả của các thuật toán học máy, trong phần thử nghiệm về dữ liệu kinh tế cụ thể là dự đoán khách hàng có đăng ký gửi tiền có kỳ hạn hay không, các thuật thuật toán học máy dưới đây đều sẽ được sử dụng để đánh giá, bao gồm:

a. Cây Quyết Định (Decision Trees),
b. Rừng Ngẫu Nhiên (Randome Forests),
c. SVM (Support Vector Machines),
d. Mạng Neural (Neural Networks),
e. k-NN (k-Nearst Neighbors),
f. Gradient Boosting.

Chúng tôi cũng sẽ sử dụng các độ đo phân loại như: Độ chính xác (Accuracy), Độ chuẩn xác (Precision), Độ nhạy (recall), Điểm F1 (F1-score) và ROC-AUC (Receiver Operating Characteristic - Area Under Curve) để đánh giá kết quả của các thuật toán khi áp dụng vào trong bài toán kinh tế.

Với bộ dữ liệu trên các mô hình học máy được triển khai bằng ngôn ngữ lập trình Python, sử dụng thư viện scikit-learn, một thư viện phổ biến cho các tác vụ học máy. Dưới đây mô tả về cách triển khai các thuật toán mà nội dung đề cập đến:

```
# Upload file CSV
from google.colab import files
uploaded = files.upload()

# Import các thư viện cần thiết
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from sklearn.metrics import accuracy_score, precision_score, recall_score,
f1_score, roc_auc_score
```

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier,
GradientBoostingClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.neural_network import MLPClassifier

# Đọc file 'BankCustomerData.csv'
file_name = 'BankCustomerData.csv'
data = pd.read_csv(file_name)

# Tách các đặc trưng và biến mục tiêu
X = data.drop(columns=['ID', 'term_deposit'])
y = data['term_deposit']

# Xác định các cột phân loại và số liệu
categorical_cols = ['job', 'marital', 'education', 'default', 'housing',
'loan', 'month']
numerical_cols = [col for col in X.columns if col not in categorical_cols]

# Tiền xử lý dữ liệu số: điền giá trị thiếu và chuẩn hóa
numerical_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='mean')),
    ('scaler', StandardScaler())
])

# Tiền xử lý dữ liệu phân loại: điền giá trị thiếu và mã hóa one-hot
categorical_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='constant', fill_value='missing')),
    ('onehot', OneHotEncoder(handle_unknown='ignore'))
])

# Kết hợp các bước tiền xử lý
preprocessor = ColumnTransformer(
    transformers=[
        ('num', numerical_transformer, numerical_cols),
        ('cat', categorical_transformer, categorical_cols)
    ])

# Chia dữ liệu thành tập huấn luyện và tập kiểm tra
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Khởi tạo một từ điển để lưu kết quả
results = {}
```

```
# Hàm để huấn luyện và đánh giá mô hình
def train_evaluate_model(model, model_name):
    pipeline = Pipeline(steps=[('preprocessor', preprocessor),
                               ('model', model)])
    pipeline.fit(X_train, y_train)
    y_pred = pipeline.predict(X_test)
    y_pred_proba = pipeline.predict_proba(X_test)[:, 1] if
hasattr(pipeline, 'predict_proba') else None
    accuracy = accuracy_score(y_test, y_pred)
    precision = precision_score(y_test, y_pred, zero_division=1)
    recall = recall_score(y_test, y_pred, zero_division=1)
    f1 = f1_score(y_test, y_pred, zero_division=1)
    roc_auc = roc_auc_score(y_test, y_pred_proba) if y_pred_proba is not
None else 'N/A'
    results[model_name] = {'Accuracy': accuracy, 'Precision': precision,
'Recall': recall, 'F1-score': f1, 'ROC-AUC': roc_auc}
    return accuracy, precision, recall, f1, roc_auc

# Decision Tree
dt_model = DecisionTreeClassifier(random_state=42)
dt_acc, dt_prec, dt_rec, dt_f1, dt_roc_auc =
train_evaluate_model(dt_model, 'Decision Tree')

# Random Forest
rf_model = RandomForestClassifier(random_state=42)
rf_acc, rf_prec, rf_rec, rf_f1, rf_roc_auc =
train_evaluate_model(rf_model, 'Random Forest')

# Support Vector Machine
svm_model = SVC(probability=True, random_state=42)
svm_acc, svm_prec, svm_rec, svm_f1, svm_roc_auc =
train_evaluate_model(svm_model, 'SVM')

# Neural Networks
nn_model = MLPClassifier(random_state=42)
nn_acc, nn_prec, nn_rec, nn_f1, nn_roc_auc =
train_evaluate_model(nn_model, 'Neural Networks')

# k-NN
knn_model = KNeighborsClassifier()
knn_acc, knn_prec, knn_rec, knn_f1, knn_roc_auc =
train_evaluate_model(knn_model, 'k-NN')

# Gradient Boosting
```

```
gb_model = GradientBoostingClassifier(random_state=42)
gb_acc, gb_prec, gb_rec, gb_f1, gb_roc_auc =
train_evaluate_model(gb_model, 'Gradient Boosting')

# Hiển thị kết quả
results_df =
pd.DataFrame(results).T.reset_index().rename(columns={'index': 'Model'})
print(results_df)
```

Sau khi huấn luyện và đánh giá các mô hình, chúng tôi nhận thấy rằng Gradient Boosting đạt được hiệu suất tốt nhất ở mục Accuracy và ROC-AUC trong việc dự đoán khách hàng có đăng ký gửi tiền có kỳ hạn hay không. Tuy nhiên ở các chỉ số đo khác như Precision, Recall, F1-Score, Gradient Boosting vẫn chưa có ra kết quả tối ưu so với các thuật toán còn lại.

Bảng 2: Kết quả so sánh mô hình

STT	Mô hình	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	ROC-AUC
1	**Decision Trees**	89.17%	42.64%	44.88%	43.73%	0.693125
2	**Random Forests**	91.58%	60.73%	29.00%	39.26%	0.921978
3	**SVM**	91.48%	64.20%	20.63%	31.22%	0.884590
4	**Neural Networks**	91.33%	54.18%	49.38%	51.67%	0.908865
5	**k-NN**	91.10%	55.30%	26.75%	36.06%	0.803699
6	**Gradient Boosting**	91.69%	60.32%	33.25%	42.87%	0.921935

Phân tích kết quả

- **Accuracy:** Gradient Boosting và Random Forests có accuracy cao nhất (91.69% và 91.58%), cho thấy khả năng dự đoán đúng tổng thể tốt. Tuy nhiên, accuracy không phải là tất cả, đặc biệt khi dữ liệu không cân bằng.
- **Precision:** SVM có precision cao nhất (64.20%), nghĩa là khi mô hình này dự đoán một mẫu là dương tính, khả năng dự đoán đúng là cao nhất.
- **Recall:** Neural Networks có recall cao nhất (49.38%), nghĩa là mô hình này có khả năng phát hiện tốt các mẫu thực sự là dương tính.
- **F1-score:** Neural Networks có F1-score cao nhất (51.67%), cho thấy sự cân bằng tốt giữa precision và recall.

- **ROC-AUC:** Gradient Boosting và Random Forests có ROC-AUC cao nhất (0.921935 và 0.921978), cho thấy khả năng phân loại tốt trên toàn bộ ngưỡng quyết định.

Kết luận:

- **Gradient Boosting và Random Forests:** Có hiệu suất tổng thể tốt nhất, thể hiện qua accuracy và ROC-AUC cao. Tuy nhiên, chúng có recall thấp hơn so với Neural Networks.
- **SVM:** Có precision cao nhất, phù hợp nếu việc giảm thiểu dương tính giả (false positives) là quan trọng. Tuy nhiên, recall của SVM thấp, có thể bỏ sót nhiều mẫu dương tính thực sự.
- **Neural Networks:** Có recall và F1-score cao nhất, phù hợp nếu việc phát hiện tất cả các mẫu dương tính là quan trọng.
- **Decision Trees và k-NN:** Có hiệu suất kém hơn so với các mô hình khác.

C. ĐÁNH GIÁ ĐỘ PHỨC TẠP THUẬT TOÁN CÀI ĐẶT

1. Thuật toán Gradient Boosting

Mục tiêu của bài toán là tìm ra thuật toán có hiệu suất tổng thế tốt nhất, cho bên phần đánh giá độ phức tạp thuật toán sẽ tập trung vào Gradient Boosting.

Gradient Boosting là một thuật toán học máy dựa trên ý tưởng kết hợp nhiều mô hình yếu (thường là các cây quyết định) để tạo ra một mô hình tối ưu hơn. Gradient Boosting hoạt động bằng cách liên tục cải thiện mô hình dự đoán thông qua việc xây dựng các mô hình yếu bổ sung. Mỗi mô hình yếu tập trung vào việc sửa chữa sai sót của mô hình trước đó, dần dần giảm thiểu sai số dự đoán và cải thiện độ chính xác của mô hình tổng thể.

Thuật toán hoạt động theo các bước sau:

Giả sử có một tập dữ liệu huấn luyện $D = \{(x_i, y_i)\}_{i=1}^{N}$

- Bắt đầu với một mô hình dự đoán ban đầu $F_0(x)$
- Với mỗi bước m từ 1 đến M:
    1. Tính các residuals (phần dư) dựa trên mô hình hiện tại $F_{m-1}(x)$:

$$r_i^{(m)} = y_i - F_{m-1}(x_i)$$

    2. Huấn luyện một mô hình yếu $h_m(x)$ để dự đoán các residuals $r_i^{(m)}$.
    3. Cập nhật mô hình dự đoán bằng cách thêm vào mô hình yếu mới:

$$F_m(x) = F_{m-1}(x) + \gamma h_m(x)$$

trong đó γ là hệ số học (learning rate).

Mô hình cuối cùng sau M bước sẽ là:

$$F_M(x) = F_0(x) + \sum_{m=1}^{M} \gamma h_m(x)$$

Áp dụng vào bài toán kinh tế cho dự đoán khách hàng gửi tiền có kỳ hạn, chúng ta có:

Input:

- x_i: Véc tơ đầu vào của các đặc trưng của khách hàng thứ i (tuổi, loại công việc, tình trạng hôn nhân, ...).
- y_i: Nhãn mục tiêu, là 1 nếu khách hàng thứ i đăng ký gửi tiền có kỳ hạn, là 0 nếu không.
- M: Số lượng mô hình yếu (cây quyết định)
- $L(y, F(x))$: Hàm mất mát, đo lường sự khác biệt giữa giá trị thực tế y và xác suất dự đoán $F(x)$.

Output:

- $F^*(x)$: Mô hình dự đoán xác suất khách hàng đăng ký gửi tiền có kỳ hạn dựa trên các đặc trưng đầu vào.

Các bước cụ thể của thuật toán:

Đầu vào:

Bước 1: Khởi tạo mô hình : Chúng ta bắt đầu với một mô hình dự đoán đơn giản, thường là xác suất trung bình của việc khách hàng đăng ký gửi tiền có kỳ hạn.

$$F_0(x) = \arg min_\gamma \sum_{i=1}^{n} L(y_i, \gamma)$$

Bước 2: Lặp lại cho $m = 1,2, \dots M$ lần

a) Tính toán giả gradient (Pseudo-residuals)
Tính giả gradient r_i^m tại mỗi bước m:

$r_i^m = -\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}$ tại $F(x) = F_{m-1}(x)$

b) Xây dựng mô hình yếu:
Huấn luyện một mô hình yếu $h_m(x)$ (thường là cây quyết định nông) để dự đoán các giả gradient r_i^m.

c) Tìm trọng số tối ưu γm cho mô hình yếu bằng cách giảm thiểu hàm mất mát trên tập huấn luyện.

$$\gamma m = argmin_{\gamma} \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + \gamma h_m(x_i))$$

d) Cập nhật mô hình: Cập nhật mô hình bằng cách thêm mô hình yếu với trọng số tối ưu vào mô hình hiện tại.
Cập nhật mô hình tại bước m

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$$

Đầu ra:

Mô hình cuối cùng $F^*(x) = F_M(x)$ sẽ được sử dụng để dự đoán xác suất khách hàng đăng ký gửi tiền có kỳ hạn dựa trên các đặc trưng đầu vào.

2. Độ phức tạp của thuật toán Gradient Boosting

Độ phức tạp về thời gian (trường hợp xấu nhất): phụ thuộc vào số lượng mẫu dữ liệu, số lượng cây và logarit của số lượng mẫu dữ liệu.

$$O(n * log(n) * số\ lượng\ ước\ tính)$$

- n: Số lượng mẫu dữ liệu (số lượng các điểm dữ liệu huấn luyện).
- $log(n)$: Hàm logarit cơ số 2 của n, thường thể hiện số lượng cây quyết định (decision trees) trong mô hình.
- $số\ lượng\ ước\ tính$: Số lượng cây trong mô hình Gradient Boosting, được xác định bởi tham số $n\_estimators$.

Độ phức tạp về không gian (trường hợp xấu nhất). Đây là một ước tính về sự tăng trưởng của bộ nhớ khi số lượng mẫu dữ liệu và số lượng cây trong mô hình tăng lên.

$$O(n * số\ lượng\ ước\ tính)$$

- n: Số lượng mẫu dữ liệu, tức là số lượng các điểm dữ liệu huấn luyện.
- $số\ lượng\ ước\ tính$: Số lượng cây trong mô hình Gradient Boosting, được xác định bởi tham số $n\_estimators$.

Áp dụng vào trong tập dữ liệu trên, chúng ta có một tập dữ liệu với 42,639 mẫu dữ liệu và muốn xây dựng một mô hình Gradient Boosting với 10 cây quyết định (estimators). Chúng ta sẽ tính toán độ phức tạp thời gian và không gian của mô hình này.

1. Số lượng mẫu dữ liệu n:

$$n = 42{,}639$$

2. Số lượng cây quyết định (số lượng ước tính):

$$số\ lượng\ ước\ tính = 10$$

2.1 Độ Phức Tạp Thời Gian

Độ phức tạp thời gian của Gradient Boosting trong trường hợp xấu nhất được biểu diễn bởi công thức:

$$O(n * log(n) * số\ lượng\ ước\ tính)$$

Tính toán từng thành phần:

$log(n)$: Đối với $n = 42{,}639, log_2(42{,}639) \approx 5{,}414$, (sử dụng logarit cơ số 2 vì cây quyết định thường là cây nhị phân).

Thay số vào công thức, chúng ta có:

$$O(42{,}639 * 5{,}414 * 10) = O(2{,}308{,}475{,}460)$$

Vậy, độ phức tạp thời gian trong trường hợp xấu nhất của mô hình Gradient Boosting với 42,639 mẫu dữ liệu và 10 cây quyết định là $O(2{,}308{,}475{,}460)$.

2.2 Độ Phức Tạp Về Không Gian

Độ phức tạp về không gian của Gradient Boosting trong trường hợp xấu nhất được biểu diễn bởi công thức:

$$O(n * số\ lượng\ ước\ tính)$$

Thay số vào công thức:

$$O(42{,}639 * 10) = O(426{,}390)$$

Vậy, độ phức tạp về không gian trong trường hợp xấu nhất của mô hình Gradient Boosting với 42,639 mẫu dữ liệu và 10 cây quyết định là $O(426{,}390)$

2.3 Tổng Kết

- Độ phức tạp thời gian (trường hợp xấu nhất): $O(2{,}308{,}475{,}460)$
- Độ phức tạp về không gian (trường hợp xấu nhất): $O(426{,}390)$

2.4 Diễn Giải

1. Thời gian: Với 42,369 mẫu dữ liệu và 10 cây quyết định, quá trình huấn luyện Gradient Boosting cần thực hiện khoảng2,308,475,460 phép tính trong trường hợp xấu nhất. Điều này cho thấy mô hình sẽ mất nhiều thời gian hơn khi số lượng mẫu dữ liệu và số lượng cây quyết định tăng lên.
2. Không gian: Với 42,369 mẫu dữ liệu và 10 cây quyết định, mô hình cần lưu trữ khoảng 426,390 thông tin trong bộ nhớ. Điều này cho thấy mô hình sẽ yêu cầu nhiều bộ nhớ hơn khi số lượng mẫu dữ liệu và số lượng cây quyết định tăng lên.

3. Chỉ số đánh giá:

Nghiên cứu này sử dụng ma trận nhầm lẫn đo độ chính xác của mô hình. Đây là phương pháp đánh giá hiệu suất phân loại các quan sát vào hai lớp có đăng ký tiền gửi (nhận giá trị 1) hay không đăng ký tiền gửi (nhận giá trị 0). Ma trận được trình bày tại Bảng 3.

Bảng 3: Ma trận nhầm lẫn

		Giá trị thực tế	
		1	0
Kết quả dự báo	1	TP	FP
	0	FN	TN

Ở đây, True Positive (TP) có nghĩa là kết quả dự báo là đúng và kết quả thực tế cũng là đúng. False Positive (FP) có nghĩa là kết quả dự báo là đúng nhưng kết quả thực tế lại là sai. Khi kết quả dự đoán là sai nhưng kết quả thực tế lại là đúng, tình huống này được gọi là False Negative (FN). Nếu kết quả dự đoán là sai và kết quả thực tế cũng là sai, điều này được gọi là True Negative (TN).

Độ chính xác (Accuracy):

$$\text{Độ } chính\ xác = \frac{TP + TN}{TP + FP + FN + TN}$$

Độ chuẩn xác (Precious):

$$\text{Độ } chuẩn\ xác = \frac{TP}{TP + FP}$$

Độ nhạy (Recall)

$$\text{Độ } nhạy = \frac{TP}{TP + FN}$$

Điểm F1 (F1-Score)

$$\text{Điểm } F1 = 2 \times \frac{Precious \times Recall}{Precious + Recall}$$

ROC và AUC

ROC Curver (Đường cong ROC). Đường cong ROC là đồ thị biểu dei6n4 m6oi1 quan hệ giữa TPR (True Positive Rate) và FPR (False Positive Rate) khi ngưỡng phân loại thay đổi.

$$TPR = \frac{TP}{TP+FN}$$

$$FPR = \frac{FP}{FP+TN}$$

AUC (Area under Curve) là diện tích dưới đường cong ROC. Nó cho biết khả năng của mô hình trong việc phân biệt giữa các lớp. Giá trị AUC dao động từ 0 đến 1.

AUC = 1 : Mô hình hoàn hảo, phân biệt hoàn toàn giữa các lớp.

AUC = 0.5 : Mô hình không tốt hơn việc dự đoán ngẫu nhiên.

AUC < 0.5 : Mô hình hoạt động kém hơn cả việc dự đoán ngẫu nhiên.

D. PHÁT TRIỂN TỪ BÀI BÁO

Dựa trên kết quả nghiên cứu của bài báo cho thấy áp dụng Gradient Boosting vào trong lĩnh vực heathcare cho ra kết quả rất tối ưu, tuy nhiên khi thực nghiệm ở dữ liệu kinh tế thì thuật toán này lại cho thấy khả năng hạn chế ở recall thấp. Đây là điểm có thể khai thác để cải thiện thuật toán này khi áp dụng cho bài toán kinh tế.

Một số hình thức có thể ứng dụng vào là triển khai các thuật toán thuộc họ Boosting như AdaBoost, XGBoosting,… cũng như cải thiện hệ số học (learning rate) sao cho phù hợp với bài toán kinh tế.

E. KẾT LUẬN

Bài nghiên cứu này đã đánh giá hiệu suất của các thuật toán học máy khác nhau trong công việc dự đoán trong lĩnh vực heathcare, dựa trên kết quả thực nghiệm ở bộ dữ liệu về kinh tế với trọng tâm là thuật toán Gradient Boosting.

Tuy nhiên, cần lưu ý rằng kết quả này chỉ mang tính tương đối và phụ thuộc vào tập dữ liệu cụ thể được sử dụng. Trong các tình huống khác, các thuật toán khác có thể đạt được hiệu suất tốt hơn.

F. TÀI LIỆU THAM KHẢO:

[1] Shah, Jaineet. (2020). Gradient Boosting.

[4] https://www.kaggle.com/datasets/tomculihiddleston/bank-customer-data-in-vietnam

[5] Samuels, Jamell. (2024). One-Hot Encoding and Two-Hot Encoding: An Introduction. 10.13140/RG.2.2.21459.76327.

[6] https://machinelearningcoban.com/tabml_book/ch_data_processing/onehot.html