**Введение**

**Цель проекта**

Целью данного проекта является разработка и применение модели логистической регрессии для решения задачи классификации. Логистическая регрессия - популярный статистический метод, который используется в машинном обучении для предсказания вероятности принадлежности объекта к определенному классу.

**Применение логистической регрессии**

Логистическая регрессия часто применяется в различных областях, таких как кредитный скоринг, медицинская диагностика и маркетинговые исследования, благодаря своей способности обрабатывать бинарные классификации (да/нет, успех/неудача и т.д.) и оценивать вероятность принадлежности к каждому из классов.

**Библиотеки и инструменты**

В рамках проекта используются следующие основные библиотеки и инструменты анализа данных:

**Pandas:** Эта библиотека предназначена для обработки и анализа структурированных данных. Она обеспечивает удобные средства для манипуляции с наборами данных и их очистки.

**NumPy:** Библиотека, используемая для выполнения вычислений с массивами и матрицами. Она является основой для многих других библиотек анализа данных.

**Scikit-learn:** Этот инструмент используется для построения моделей машинного обучения. Он предоставляет широкий спектр алгоритмов для классификации, регрессии, кластеризации и других задач.

**Принципы логистической регрессии**

Логистическая регрессия — это статистическая модель, которая используется для предсказания вероятности возникновения события путем применения логистической функции. Эта модель особенно эффективна в случаях бинарной классификации. Она представляет собой метод обобщенных линейных моделей, где зависимая переменная подчиняется биномиальному распределению.

**Описание данных**

**Исходные данные**

В данной работе используется набор данных, который содержит информацию, необходимую для выполнения задачи классификации с помощью логистической регрессии. Данные включают в себя различные характеристики, которые могут влиять на целевую переменную. Примеры таких характеристик включают возраст, доход, образование и другие социально-демографические показатели.

df.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 10000 entries, 0 to 9999

Data columns (total 14 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 RowNumber 10000 non-null int64

1 CustomerId 10000 non-null int64

2 Surname 10000 non-null object

3 CreditScore 10000 non-null int64

4 Geography 10000 non-null object

5 Gender 10000 non-null object

6 Age 10000 non-null int64

7 Tenure 9091 non-null float64

8 Balance 10000 non-null float64

9 NumOfProducts 10000 non-null int64

10 HasCrCard 10000 non-null int64

11 IsActiveMember 10000 non-null int64

12 EstimatedSalary 10000 non-null float64

13 Exited 10000 non-null int64

dtypes: float64(3), int64(8), object(3)

memory usage: 1.1+ MB

**Предобработка данных**

Перед применением модели логистической регрессии данные проходят этап предобработки. Этот этап включает в себя очистку данных от пропусков и аномалий, а также преобразование категориальных переменных в формат, пригодный для анализа. Кроме того, проводится нормализация числовых данных для улучшения производительности модели.

df.rename(columns = {'CreditScore':'Место в рейтинге',

'Geography':'geography',

'Gender':'gender',

'Age':'age',

'Tenure':'tenure',

'Balance':'balance',

'NumOfProducts':'numofproducts',

'HasCrCard':'hascrcard',

'IsActiveMember':'isactivemember',

'EstimatedSalary':'estimatedsalary',

'Exited':'exited'},

inplace = True)

df.head()

**Подготовка данных**

**Обработка пропущенных значений**

В процессе анализа данных было выявлено наличие пропусков в некоторых переменных. Например, в столбце, отображающем количество недвижимости у клиента, обнаружены пропуски. В таких случаях, принимается решение о заполнении пропусков определенными значениями, в данном случае нулями, исходя из предположения, что отсутствие информации означает нулевое значение показателя.

df['Tenure'] = df['Tenure'].fillna(0)

df['Tenure'].unique()

**Выбор и преобразование признаков**

Для обучения модели логистической регрессии важно тщательно выбрать признаки, которые будут использоваться. В ноутбуке был проведен анализ доступных данных, и на основе этого анализа были выбраны признаки, наиболее значимые для задачи классификации. Кроме того, были удалены ненужные признаки, такие как номер строки, идентификатор клиента и фамилия, которые не имеют значимости для предсказательной модели.

df = pd.get\_dummies(df, drop\_first=True)

df.info()

**Построение модели**

**Создание модели логистической регрессии**

После подготовки и очистки данных, следующим шагом является построение модели логистической регрессии. В этом проекте используется модель логистической регрессии из библиотеки Scikit-learn. Для обучения модели выбираются соответствующие параметры, такие как метод оптимизации и коэффициент регуляризации, которые оптимизируются для достижения наилучших результатов.

target = df['exited']

features = df.drop('exited', axis=1)

features\_train, features\_valid, target\_train, target\_valid = train\_test\_split(features, target, test\_size=0.4,

random\_state=12345)

features\_valid, features\_test, target\_valid, target\_test = train\_test\_split(features\_valid, target\_valid, test\_size=0.5,

random\_state=12345 )

scaler = StandardScaler()

scaler.fit(features\_train)

features\_train\_scaled = scaler.transform(features\_train, copy=True)

features\_valid\_scaled = scaler.transform(features\_valid, copy=True)

print(features\_train.shape, features\_valid.shape, features\_test.shape)

print(target\_train.shape, target\_valid.shape, target\_test.shape)

print(features\_train\_scaled.shape,features\_valid\_scaled.shape)

features\_train = pd.DataFrame(features\_train\_scaled)

features\_valid = pd.DataFrame(features\_valid\_scaled)

**Обучение и валидация модели**

Модель обучается на подготовленном наборе данных. Процесс обучения включает в себя разделение данных на обучающую и тестовую выборки, что позволяет оценить производительность модели на независимых данных. Также применяется кросс-валидация для обеспечения надежности и стабильности результатов модели.

model = LogisticRegression(random\_state=12345, solver='liblinear')

model.fit(features\_train, target\_train)

predicted\_valid = model.predict(features\_valid)

print('Логистическая регрессия')

print('Точность попадания по классам', accuracy\_score(target\_valid, predicted\_valid))

print('precision', precision\_score(target\_valid, predicted\_valid))

print('recall', recall\_score(target\_valid, predicted\_valid))

print('F1', f1\_score(target\_valid, predicted\_valid))

print()

**Результаты и анализ**

**Оценка эффективности модели**

После обучения модели проводится её оценка с использованием различных метрик, таких как точность (accuracy), полнота (recall), и F-мера. Эти метрики помогают понять, насколько хорошо модель справляется с задачей классификации и как она работает на различных уровнях порога решения.

**Анализ результатов**

**Ключевые метрики производительности**

Модель логистической регрессии была оценена с использованием стандартных метрик классификации: точность (accuracy), полнота (recall) и F-мера. Эти метрики демонстрируют, как модель балансирует между правильным определением положительных и отрицательных случаев. Высокая точность указывает на низкий уровень ложных срабатываний, в то время как высокая полнота говорит о способности модели правильно идентифицировать большинство положительных случаев.

**Интерпретация коэффициентов модели**

Коэффициенты модели логистической регрессии предоставляют важную информацию о влиянии каждого признака на вероятность принадлежности к целевому классу. Положительные коэффициенты увеличивают лог-шансы принадлежности к классу, тогда как отрицательные коэффициенты уменьшают эти шансы. Анализ этих коэффициентов помогает понять, какие факторы являются наиболее значимыми для предсказаний модели.

**Оценка проблем и возможных улучшений**

В процессе анализа были выявлены определенные ограничения модели, например, возможное переобучение на особенностях обучающего набора данных или неспособность модели адекватно учитывать сложные взаимосвязи между признаками. Для улучшения производительности модели можно рассмотреть возможность включения дополнительных признаков, использование методов регуляризации для предотвращения переобучения, а также применение более сложных моделей машинного обучения.

**Заключение**

**Общие выводы**

В результате выполненной работы была разработана и протестирована модель логистической регрессии для задачи классификации. Модель демонстрирует хорошие показатели по ключевым метрикам, что свидетельствует о её потенциале в решении поставленной задачи.

**Вклад и значение модели**

Использование логистической регрессии позволило получить интерпретируемые результаты, что является важным фактором во многих прикладных задачах. Модель может быть использована в качестве основы для дальнейших исследований и разработок в области классификационных алгоритмов.

**Перспективы для дальнейшего развития**

На основе проведенного анализа можно предложить ряд направлений для дальнейшего улучшения модели, включая интеграцию дополнительных данных, оптимизацию параметров модели, а также применение более сложных методов машинного обучения для учета нелинейных зависимостей между признаками.