5. Шебеко К.К., Шебеко Д.К. Институциональное доверие как фактор экономического развития: эмпирический анализ // Экономика и банки. 2020. №2. URL: https://cyberleninka.ru/article/n/institutsionalnoe-doverie-kak-faktor-ekonomicheskogo-razvitiya-empiricheskiy-analiz (дата обращения: 09.11.2021).

УДК 004.021

В.К. Шешолко, И.А. Старовойтов
Академия управления при Президенте Республики Беларусь,
Минск, Беларусь

ПОСТРОЕНИЕ БАЗОВЫХ МОДЕЛЕЙ МАШИННОГО ОБУЧЕНИЯ НА ЯЗЫКЕ PYTHON

Аннотация. *Показана последовательность шагов разработки модели с использованием библиотек языка Pyton на примере решения задачи классификации.*

Результат работы модели будет зависеть от правильности настройки параметров и качества обучающих данных. Данная модель может в дальнейшем использоваться для оценки неизвестных данных.

V.K. Shesholko, I.A. Starovoitov
Academy of Public Administration
under the aegis of the President of the Republic of Belarus,
Minsk, Republic of Belarus

DEVELOPMENT OF BASIC MODELS OF MACHINE LEARNING IN PYTHON LANGUAGE

Abstract. *The sequence of steps for developing a model using the Pyton language libraries is shown on the example of solving the classification problem.*

The result of the model will depend on the correct setting of the parameters and the quality of the training data. This model can then be used to estimate unknown data.

1.Общее понятие о машинном обучении

На сегодняшний день Python – один из лидеров среди языков, предназначенных для создания моделей машинного обучения. Ежедневно десятки тысяч разработчиков создают сотни различных моделей именно благодаря этому языку. Ближайшей альтернативой Python является Go (Golang) - многопоточный язык программирования, созданный компанией

Google в 2009 году. Сам по себе данный язык предназначен для создания различного рода приложений (веб-сервисы и клиент-серверные приложения). Хотя этот язык также обладает возможностями по работе с графикой и машинным обучением. Если рассматривать эти два языка с точки зрения работы с машинным обучением, то для этого больше подходит язык Python, ведь именно он имеет большее разнообразие различных библиотек, которые постоянно совершенствуются.

Для начала работы с машинным обучением в первую очередь необходимо сформулировать задачу, которую необходимо решить. Важное значение имеет выбор среды, где будет написан и выполнен код. Для работы с языком Python большинство программистов предпочитают блокноты-файлы с расширением .ipynb, которые представляют собой документы с возможностью написания и выполнения в нем кода (рис. 1). В сети существует огромное количество различных сайтов, которые позволяют бесплатно создать и редактировать блокноты, однако самым популярным местом является Google Colab Laboratory (https://colab.research.google.com).

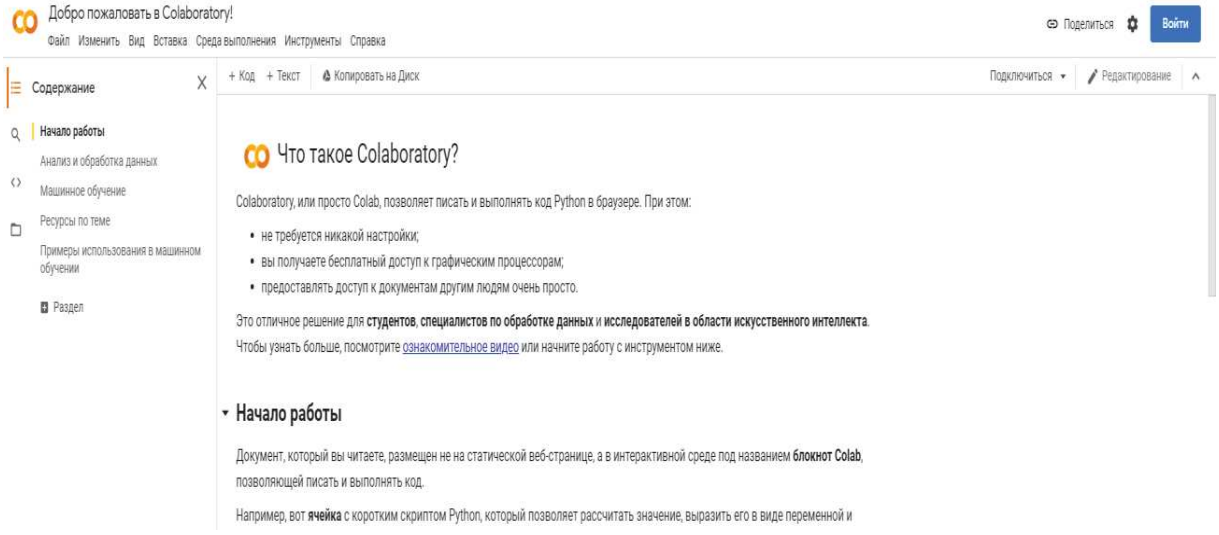

Рис. 1 - Начальная страница Google Colab

Как можно увидеть, Colab позволяет выполнять любой код Python прямо в браузере, подключившись для этого как к локальной, так и к удаленной среде выполнения, в зависимости от желания программиста (рис. 2).

```
[ ]  seconds_in_a_day = 24 * 60 * 60
     seconds_in_a_day

     86400
```

Рис. 2 - Запись в ячейке с кодом Python

После выполнения кода в ячейке, созданной пользователем, вывод, в случае необходимости, будет представлен ниже выделенной ячейки. Colab – многофункциональная система, самые базовые функции – чтение и ввод.

2.Построение модели

Для построения модели будет использоваться библиотека *scikit-learn* – одна из самых простых и популярных способов создать модель, используя минимальное количество строк. Для обучения модели будет взят набор данных (data set) «Titanic» – информация о пассажирах экипажа, которые бронировали себе места на одноименном корабле. Указанный выше data set полностью отсортирован от «мусорных» данных и содержит минимальное количество пустых ячеек. В реальном мире такие данные встречаются крайне редко, именно поэтому перед построением и обучением модели необходимо провести *анализ данных* – изучение данных перед их использованием. Данное действие очень часто позволяет обнаружить закономерности или аномалии в данных, что может помочь в обучении, а также очистить данные перед использованием. В статье этап анализа данных опущен.

Если задача поставлена, например задача классификации, построение модели начинается с подключения необходимых библиотек (рис. 3). На этом этапе необходимо выбрать оптимальный метод решения задачи. Выбрав алгоритм случайного леса (RandomForestClassifier) в качестве метода классификации, стоит оценить его достоинства и недостатки.

Алгоритм случайного леса (Random Forest) – универсальный алгоритм машинного обучения, суть которого состоит в использовании ансамбля решающих деревьев. Имеет высокую точность предсказания, не требует тщательной настройки параметров, практически не чувствителен к выбросам в данных, редко переобучается, способен эффективно обрабатывать данные с большим числом признаков и классов, одинаково хорошо обрабатывает как непрерывные, так и дискретные признаки.

Некоторые из пометок к коду будут указаны в виде комментариев непосредственно в самом коде.

```
[53] import pandas as pd # Библиотека pandas (импортированная под именем pd) для взаимодействия с датасетом Titanic
     from sklearn.ensemble import RandomForestClassifier # Классификатор, импортированный из подразделения ensemble библиотеки sklearn
```

Рис. 3- Подключение сторонних библиотек

После подключения библиотек необходимо внести файлы с данными для их дальнейшего использования (рис. 4).

```
[54] train_data = pd.read_csv("/content/train.csv") # Импорт тренировочного датасета
     test_data = pd.read_csv("/content/test.csv") # Импорт тестировочного датасета
```

Рис. 4 - Подключение набора данных Titanic

Для начала работы с данными их необходимо разбить по принципу «значения - ключи». Для этого выделим искомый столбец и запишем его в переменную «y», а столбцы, которые мы будем использовать для обучения – в переменную «X» (рис. 5)

```
features = ["Pclass", "SibSp", "Parch"] # Указание необходимых стобцов
X = train_data[features] # Запись толбцов как X
y = train_data['Survived'] # Искомый столбец
X_test = test_data[features] # Получение столбцов из тестового набора данных
```

Рис. 5 - Разбивка столбцов на переменные для обучения модели

Для обучения модели необходимо ее создать и привязать ее к переменной – это позволит работать с моделью, изменяя ее свойства (рис. 5). После создания модели программа сообщает все настройки, имеющиеся у данного экземпляра модели и их значения. Например, параметр max_features, который определяет максимальное количество признаков, которые будут рассматриваться на каждом узле; параметр bootstrap позволяющий задать, будет ли подмножество наблюдений, рассматриваемых для дерева, создаваться с использованием выборки с возвратом (настройка по умолчанию) либо без возврата; n_estimators, который задает количество деревьев решений для включения в лес. После инициализации можно приступать к обучению модели. Для этого используется метод «.fit()», который принимает в качестве аргументов тренировочные данные «X» и «y».

```
[63] model = RandomForestClassifier() #Инициализация модели
     model.fit(X,y) # Обучение модели на основе данных
```

```
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                       criterion='gini', max_depth=None, max_features='auto',
                       max_leaf_nodes=None, max_samples=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=100,
                       n_jobs=None, oob_score=False, random_state=None,
                       verbose=0, warm_start=False)
```

Рис. 5 - Инициализация модели и получение ее настроек

На следующем этапе необходимо узнать, какие признаки наиболее важны в модели случайного леса. В библиотеке scikit-lean классификационные случайные леса могут сообщать об относительной важности каждого признака с помощью атрибута feature_importances.

Для нашего примера обычно требуется вручную определить структуру и размер дерева принятия решений. Древесные обучающиеся алгоритмы библиотеки scikit-lean имеют множество приемов управления размером дерева принятия решений. Доступ к ним осуществляется через следующие параметры: max_depth – максимальная глубина дерева; min_samples_split – минимальное количество наблюдений в узле прежде, чем этот узел будет расщеплен; min_samples_leaf – минимальное количество наблюдений, которое должно находиться в листе; max_leaf_nodes – максимальное количество листьев; min_impurity_split – минимальное требуемое уменьшение разнородности прежде, чем расщепление будет выполнено.

На следующем этапе следует оценить качество полученной модели.

Для каждого дерева модели существует отдельное подмножество наблюдений, которые не используются для тренировки этого дерева. Они называются *внепакетными наблюдениями* (out-of-bag, OOB). Их можно использовать в качестве тестового набора для оценки результативности нашего случайного леса. Для каждого наблюдения обучающийся алгоритм сравнивает истинное значение наблюдений с предсказанием из подмножества деревьев, не натренированных с помощью этого наблюдения. Вычисляемая общая оценка предоставляет единую меру результативности случайного леса.

В библиотеке scikit-leam можно выполнять внепакетное оценивание случайного леса, задав в объекте случайного леса (т. е. объекте класса RandomForestCiassifier) параметр oob_score=True. Значение оценки может быть получено с помощью атрибута oob_score_.

Разработанная модель может в дальнейшем использоваться для оценки неизвестных данных.

```
[64] predictions = model.predict(X_test) # Получение предсказаний для неизвестных модели данных
```

Рис. 6 - Предсказывание модели на основе имеющихся данных

Результаты работы модели будут зависеть от правильности настройки параметров и качества обучающих данных.

Список использованных источников

1. Введение в язык Go.- Режим доступа https://metanit.com/go/tutorial/1.1.php.- Дата доступа 25.11.2021

УДК 347.65

В.В. Шпаковская
Нотариальное бюро № 1 г. Бобруйска
Бобруйск, Беларусь

НАСЛЕДОВАНИЕ ЦИФРОВЫХ АКТИВОВ КАК ОБЪЕКТОВ ГРАЖДАНСКИХ ПРАВ

Аннотация. *Рассматривается вопрос о включении цифровых активов в объекты гражданских прав как «иное имущество». Анализируется возможность наследования цифровых активов как по закону, так и по завещанию. Предлагается осуществлять наследование токенов не только по завещанию, но и по закону.*

V.V. Shpakovskaya
Notary office No. 1 of Bobruisk,
Bobruisk, Belarus

INHERITANCE OF DIGITAL ASSETS AS OBJECTS OF CIVIL RIGHTS