## Общие положения о предоставлении решения задач

Представленные далее задачи должны быть решены с помощью скриптов, написанных на языке Python с использованием библиотек, доступных в стандартных репозиториях.

Документ, содержащий решения задач, должен представлять собой файл формата Microsoft Word, содержащий следующую информацию для каждой задачи:

- формулировка текущей задачи (без описания входных данных). Допускается копирование текста задания из настоящего документа;

- общие положения о решении пунктов задачи. Для каждого пункта кратко описываются основные теоретические и практические положения, описание и обоснование использованных методов решения текущего пункта задачи. Если в тексте задания или пункте задания требуется вывести некоторые данные в консоль или составить рисунок, данные элементы вывода встраиваются в текст текущего пункта. Допускается вставка текста из консоли/вывода путем обрезки скриншотов с требуемой информацией. При требовании вывода полученных в ходе выполнения таблиц, если в таблице более 5 строк, допускается приведение первых пяти строк таблицы. Данные, выводимые в консоль/в файл в текстовом формате, должны быть подписаны;

- в приложении к задаче после всего документа (после всех общих положений о решении задач) указывается полный код программы/скрипта, использованного для решения задачи. Код должен быть снабжен комментариями для понимания процесса решения, предпочитаемый язык комментариев – английский. Для удобства приложение озаглавить как «Приложение к задаче [N]».

## Задача 1 (общая работа с Python, бинарная классификация, библиотеки pandas, numpy, scipy, matplotlib).

**Входные данные.** В файле Problem1.xlsx содержится таблица данных о работе службы доставки магазина электронной коммерции. Структура таблицы представляет собой следующий набор полей:

| ID | Идентификатор записи в таблице |
| --- | --- |
| Warehouse\_block | Обозначение блока склада, с которого производилась доставка |
| Mode\_of\_Shipment | Метка метода доставки |
| Customer\_care\_calls | Количество звонков, совершенных для запроса отгрузки со склада |
| Customer\_rating | Рейтинг клиента |
| Cost\_of\_the\_Product | Цена продукта |
| Prior\_purchases | Количество предыдущих покупок клиента |
| Product\_importance | Внутренняя маркировка магазина о степени приоритетности |
| Gender | Пол клиента |
| Discount\_offered | Предложенная скидка на покупку |
| Weight\_in\_gms | Вес посылки |
| Reached.on.Time\_Y.N | Флаг, отвечающий за факт своевременной доставки |
| proba1 | Вероятность своевременной доставки, полученная классификатором 1 |
| proba2 | Вероятность своевременной доставки, полученная классификатором 2 |

Флаг своевременной доставки в данной задаче является целевым, далее так же будут обозначаться соответствующее ему поле и столбец. Вероятности своевременной доставки, полученные классификаторами 1 и 2, не являются объясняющими переменными для целевой переменной в данной задаче.

**Задача.**

Задание должно быть реализовано с помощью библиотек pandas, numpy, scipy и стандартных средств Python.

1. Сформировать DataFrame с исходными данными.
2. Вычислить и вывести в консоль коэффициенты попарной корреляции между каждым из объясняющих полей и целевым полем. Соотнести полученные коэффициенты корреляции и экономический/логический смысл взаимоотношений между величинами, указанными в полях, и целевой величиной.
3. Вычислить среднее значение для каждого из числовых объясняющих полей в выборке. Для текстовых полей вычислить наиболее распространенное значение. На основе полученных данных скомпоновать и вывести в консоль «типичные» параметры доставки для данной выборки.
4. Для каждого классификатора, для которых указана прогнозируемая вероятность своевременной доставки, построить ROC-кривую. Графики вывести в консоль.
5. Вычислить и вывести в консоль коэффициенты AUC для каждого классификатора. Сравнить классификаторы на основе полученных данных, сделать выводы.
6. Вычислить и вывести в консоль значения других доступных в библиотеке scipy метрик качества бинарной классификации (не менее двух). Сравнить классификаторы на основе этих метрик, сделать выводы.

## Задача 2 (общая работа с Python, типы данных, ООП, библиотека pandas).

**Входные данные.** В файле Problem2.csv содержится таблица данных о результатах игр лиги NBA за сезон 2020-2021 года (на момент составления задания). Структура таблицы представляет собой следующий набор полей:

| date | Дата проведения игры |
| --- | --- |
| team1 | Трехбуквенное (кодовое) название команды 1 |
| team2 | Трехбуквенное (кодовое) название команды 2 |
| elo\_prob1 | Вычисленная по методике ELO априорная вероятность победы в матче для команды 1 |
| elo\_prob2 | Вычисленная по методике ELO априорная вероятность победы в матче для команды 2 |
| score1 | Количество очков, набранное командой 1 |
| score2 | Количество очков, набранное командой 2 |
| \* | Неиспользуемые столбцы |

Столбцы, явно не указанные в таблице выше, не используются в текущем задании.

**Задача.**

Задание должно быть реализовано с помощью библиотек pandas и стандартных средств Python.

1. Предобработка: на основе исходных данных сформировать DataFrame без неиспользуемых столбцов и содержащий только уже прошедшие игры. Вывести в консоль измерения (кол-во строк х кол-во столбцов) полученного DataFrame до и после предобработки.
2. Реализовать класс (объект) Team, содержащий два поля – строковое имя команды и список игр этой команды. Список игр команды должен быть типа «список» и нести в себе экземпляры класса Game, относящиеся к данной команде (объект Game описан далее);
3. Реализовать класс Game, описывающий данные об одной игре между командой, указанной в соответствующем экземпляре класса Team, и командой-оппонентом. Класс Game должен содержать поля:
   1. Дата игры;
   2. Вероятность победы команды, к которой относится данный экземпляр
   3. Количество очков команды, к которой относится данный экземпляр
   4. Имя команды-противника
   5. Вероятность победы команды–противника в отдельно взятой игре
   6. Количество очков команды–противника в отдельно взятой игре
   7. Автоматически вычисляемое булево поле, означающее победу (True) или поражение (False) команды, к которой относится данный экземпляр

*Например, если по исходным данным 22.12.2020 произошла игра между командами BRK и GSW, закончившаяся со счетом 125:99 (для упрощения в примере опущены вероятности выигрыша), то в экземпляре класса Team, относящегося к команде BRK, помимо имени BRK должен находиться список с экземпляром класса Game, в котором указаны поля даты игры как 22.12.2020, очков команды как 125, имени команды-противника как GSW, очков команды-противника как 99. Также автоматически должно быть вычислено значение поля победы (в данном случае команда BRK выиграла, поэтому оно должно быть обозначено как True)*

1. Используя предобработанные данные, сформировать словарь экземпляров класса Team для каждой команды. Ключами словаря должны быть краткие имена команд. При формировании для каждой команды заполнить список игр с экземплярами класса Game.
2. В классе Game реализовать метод вывода в консоль данных игры, содержащий следующие данные: буквенное обозначение выигрыша или проигрыша, имя команды-противника, дата игры, счет игры. Например:

*L vs BRK; date 2020-12-22; score 99:125*

После реализации вывести в консоль данные первой игры для команд GSW и DAL. (Под первой игрой здесь подразумевается игра, находящаяся в начале сформированного списка игр).

1. В классе Team реализовать метод вывода в консоль всех игр команды. Вывести в консоль данные всех игр команд LAС и NYK.
2. В классe Team реализовать метод вывода в консоль всех игр команды, в которых команда проиграла, а вероятность победы команды была больше вероятности победы команды-противника. Вывести в консоль результат работы метода для команд СHI и BOS.
3. В классe Team реализовать метод вывода в консоль всех игр команды, в которых команда выиграла, а вероятность победы команды была меньше вероятности победы команды-противника. Вывести в консоль результат работы метода для команд LAL и UTA.

**Указание для заданий пунктов 6, 7, 8.** Для вывода данных одной игры необходимо использовать реализованный ранее метод показа данных игры класса Game. Перед выводом данных об играх рекомендуется вывести сообщение о команде, для которой вызывается метод и краткое обозначение метода. Также подразумевается, что при необходимости в классе Game можно реализовывать вспомогательные методы, на которые будут опираться методы класса Team.

## Задача 3 (общая работа с Python, работа с файловой системой, обработка строк, библиотека pandas, matplotlib).

**Входные данные.** В архиве Problem3.7z содержится папка Problem3 с файлами, в которых содержатся данные с текстами песен некоторых популярных исполнителей.

Предварительно архив должен быть распакован в директорию с файлами скриптов. Название файлов соответствует формату

*[краткое имя исполнителя].[расширение]*

Требуемые файлы имеют одно из следующих расширений: .csv, .xlsx, .bindata. Файлы формата .bindata представляют собой запись байтового потока оригинальных таблиц типа DataFrame. В файлах расположены таблицы следующей структуры:

| Artist | Псевдоним исполнителя |
| --- | --- |
| Title | Название песни |
| Album | Название альбома |
| Year | Год выпуска альбома |
| Date | Дата выпуска альбома/песни |
| Lyric | Текст песни |

**Задача.**

Задание должно быть реализовано с помощью библиотек pandas, matplotlib и стандартных средств Python.

1. Реализовать считывание файлов, находящихся в папке Problem3. Для каждого файла в папке с расширением, входящим в список требуемых, реализовать считывание данных и формирование DataFrame. Все считанные DataFrame поместить в словарь со значениями-ключами, взятыми как краткие имена исполнителей из названий соответствующих файлов, т.е. словарь должен соответствовать отношению «краткое имя исполнителя из имени файла» - «считанный из соотв. файла Dataframe». Если расширение файла не входит в список расширений требуемых файлов, вывести сообщение в консоль о невозможности чтения файла с указанием имени файла или его расширения.

**Указание.** Чтение файлов формата .bindata реализовать через байтовое чтение файла с использованием модуля pickle.

1. Пользуясь полученным словарем данных, вывести в консоль список имен исполнителей, для которых данные были прочитаны. Объединить все считанные DataFrame для разных исполнителей в один объект DataFrame. Вывести в консоль размеры (кол-во строк х кол-во столбцов) получившегося DataFrame.
2. В полученном DataFrame на основе столбца текста песни сформировать новый столбец, содержащий количество слов в каждой песне. Словом считается последовательность букв и символов, разделенная пробелом, знаком препинания или непечатаемым разделительным символом. Если столбец с текстом песни для какой-либо записи пуст, присвоить значению формируемого столбца значение 1. Вывести в консоль список длин всех песен в словах исполнителя Eminem; для исполнителя Coldplay вывести список длин песен, вышедших в 2018 году.
3. В полученном DataFrame на основе столбца текста песни сформировать новый столбец, содержащий самое длинное слово песни. Если столбец с текстом песни для какой-либо записи пуст, присвоить значению формируемого столбца значение пустой строки. Вывести в консоль список самых длинных слов в песнях исполнителя Post Malone; для исполнителя Beyonсé вывести список самых длинных слов в песнях с альбома BEYONСÉ.
4. На основе полученного DataFrame вывести графики с динамикой изменения средней длины песен в словах по годам для исполнителей Rihanna и Maroon 5. Все записи, в которых не указан год выпуска песни/альбома, или для которых значение в столбце с количеством слов равно 1, не должны учитываться. На основании полученных графиков сделать выводы о характере изменений среднего количества слов в песнях во времени.