Знакомство с языком Python (семинары)

Задание 1. one hot

В ячейке ниже представлен код генерирующий *DataFrame*, которая состоит всего из 1 столбца. Ваша задача перевести его в *one hot* вид. Сможете ли вы это сделать без `get_dummies`?

```
Unset
import random
lst = ['robot'] * 10
lst += ['human'] * 10
random.shuffle(lst)
data = pd.DataFrame({'whoAmI':lst})
data.head()
```

Подсказка № 1

Перед началом работы убедитесь, что у вас есть исходный DataFrame с нужным столбцом. Если DataFrame еще не создан, выполните все необходимые шаги для его создания.

Подсказка № 2

Получите уникальные значения в столбце, который вы хотите перевести в формат one-hot. Эти уникальные значения будут использоваться для создания новых столбцов в вашем DataFrame. Используйте метод `unique()` для извлечения всех уникальных значений из столбца. Эти значения станут заголовками новых столбцов в one-hot кодировании.

Подсказка № 3

Создайте новый DataFrame для хранения one-hot кодирования. Для каждого уникального значения в исходном столбце создайте новый столбец в этом DataFrame. Инициализируйте пустой DataFrame, а затем добавляйте в него столбцы с one-hot кодированием для каждого уникального значения.

Подсказка № 4

Для каждого уникального значения в исходном столбце создайте бинарный столбец в новом DataFrame. Этот столбец должен содержать 1, если значение в строке совпадает с текущим уникальным значением, и 0 в противном случае. Используйте

условное выражение `(data['whoAmI'] == value)` для создания бинарных столбцов. Преобразуйте результат в тип `int`, чтобы получить 0 или 1.

Подсказка № 5

Объедините DataFrame с one-hot кодированием с исходным DataFrame, если необходимо. Это позволит вам видеть исходные данные вместе с их one-hot представлением. Используйте `pd.concat()` для объединения DataFrame. Убедитесь, что вы объединяете по столбцам (`axis=1`), чтобы добавить новые столбцы к существующему DataFrame.

Эталонное решение:

```
import pandas as pd

import random



# Генерация DataFrame

lst = ['robot'] * 10

lst += ['human'] * 10

random.shuffle(lst)

data = pd.DataFrame({'whoAmI': lst})



# Создание one-hot кодирования

unique_values = data['whoAmI'].unique()  # Находим уникальные
значения

one_hot = pd.DataFrame()



for value in unique_values:

    one_hot[value] = (data['whoAmI'] == value).astype(int)



# Объединение one-hot кодирования с исходным DataFrame (опционально)

data = pd.concat([data, one_hot], axis=1)
```

```
print(data.head())
```

Задание 2. Анализ расходов по возрасту.

Постройте линейный график, где по оси X будет отображаться возраст (age), а по оси Y — балл по расходам (spending_score). Этот график поможет визуализировать, как изменяются расходы в зависимости от возраста сотрудников. Проанализируйте тренды и выявите возможные закономерности.

Подсказка № 1

Проверьте наличие и корректность данных. Убедитесь, что в столбцах age и spending_score есть данные, и они не содержат пропусков. Вы можете использовать метод df.info() для проверки наличия пустых значений и метод df.head() для предварительного просмотра данных.

Подсказка № 2

Проверьте, что данные в столбцах `age` и `spending_score` имеют правильные типы данных (например, числовые значения). Если данные представлены в виде строк, используйте методы `pd.to_numeric()` или `astype()` для преобразования в числовые значения.

Подсказка № 3

Создайте график с помощью Seaborn**.** Используйте метод `sns.lineplot()` для построения линейного графика. Убедитесь, что вы правильно указали параметры `x` и `y` для осей графика, передавая соответствующие имена столбцов из DataFrame.

Подсказка № 4

Настройте оформление графика. Добавьте заголовок, метки осей и при необходимости настройте стиль графика. Используйте `plt.title()` для заголовка, `plt.xlabel()` для метки оси X, и `plt.ylabel()` для метки оси Y. Это сделает ваш график более понятным и информативным.

Эталонное решение:

```
import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt



# Загрузка данных
```

```
df = pd.read_csv('data.csv')



# Проверка названий столбцов

print(df.columns)



sns.lineplot(x='age', y='spending_score', data=df)

plt.title('Age vs Spending Score')

plt.xlabel('Age')

plt.ylabel('Spending Score')

plt.show()
```

Задание 3. Взаимосвязь между зарплатой и бонусами

Создайте точечный график, где по оси X будет отображаться зарплата (`salary`), а по оси Y — бонусы (`bonus`). Размер точек на графике должен быть пропорционален количеству лет в компании (`years_at_company`). Этот график позволит исследовать взаимосвязь между зарплатой и бонусами и оценить влияние стажа на размер бонусов.

Подсказка № 1

Проверьте, что в столбцах `salary`, `bonus` и `years_at_company` отсутствуют пропущенные значения. Пропуски могут негативно повлиять на визуализацию. Используйте метод `df.isnull().sum()` для проверки наличия пропусков в столбцах и метод `df.dropna()` для их удаления, если необходимо.

Подсказка № 2

При создании точечного графика с `sns.scatterplot()`, используйте параметр `size` для изменения размера точек в зависимости от количества лет в компании (`years_at_company`). Это поможет визуально представить, как стаж влияет на размер бонусов.

Подсказка № 3

Проверьте, что размер точек на графике не слишком маленький или большой. Вы можете настроить размер точек с помощью параметра `sizes` в функции `sns.scatterplot()`, чтобы они были более читаемыми.

Подсказка № 4

Добавьте заголовок, метки осей и настройте стиль графика. Используйте функции `plt.title()`, `plt.xlabel()`, и `plt.ylabel()` для улучшения визуального представления графика и легкости интерпретации.

Эталонное решение:

```
import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt



# Загрузка данных из CSV-файла

df = pd.read_csv('data.csv')



# Создание точечного графика с использованием Seaborn

# Ось X: зарплата (salary)

# Ось Y: бонусы (bonus)

# Размер точек пропорционален количеству лет в компании
(years_at_company)

sns.scatterplot(x='salary', y='bonus', size='years_at_company',
data=df)



# Настройка заголовка графика

plt.title('Salary vs Bonus with Years at Company')



# Настройка меток осей

plt.xlabel('Salary')

plt.ylabel('Bonus')



# Отображение графика
```

```
plt.show()
```