Project Title	**coffee sales**
Tools	ML, SQL, Excel
Technologies	Data Analyst & Data scientist
Project Difficulties level	intermediate

Dataset : Dataset is available in the given link. You can download it at your convenience.

Click here to download data set

About Dataset

Overview

This dataset contains detailed records of coffee sales from a vending machine.
The vending machine is the work of a dataset author who is committed to providing an open dataset to the community.
It is intended for analysis of purchasing patterns, sales trends, and customer preferences related to coffee products.

Data Collection Period

The dataset spans from March 2024 to Present time, capturing daily transaction data. And new information continues to be added.

Tasks

- Time Series Exploratory Data Analysis
- Next day/week/month sales
- Specific customer purchases

Author

Yaroslav Isaienkov @ihelon

NOTE :
1. this project is only for your guidance, not exactly the same you have to create. Here I am trying to show the way or idea of what steps you can follow and how your projects look. Some projects are very advanced (because it will be made with the help of flask, nlp, advance ai, advance DL and some advanced things) which you can not understand .
2. You can make or analyze your project with yourself, with your idea, make it more creative from where we can get some information and understand about our business. make sure what overall things you have created all things you understand very well.

Sure! Below is a step-by-step guide to performing a coffee sales analysis using Python, focusing on data cleaning and basic machine learning (ML) modeling. This example uses pandas for data manipulation and scikit-learn for machine learning. I'll assume you have a dataset named `coffee_sales.csv`.

1. Data Collection

First, ensure you have the necessary libraries installed:

bash

Copy code

```
pip install pandas scikit-learn matplotlib seaborn
```

2. Data Preparation and Cleaning

Load and inspect the data:

```
import pandas as pd

# Load the dataset
data = pd.read_csv('coffee_sales.csv')

# Display the first few rows
print(data.head())
```

Assume the dataset has the following columns: `Date`, `Store`, `Product`, `Sales`, `Quantity`, `Price`.

Handling Missing Values

```
# Check for missing values
print(data.isnull().sum())

# Fill missing numerical values with the median
```

```
data['Sales'].fillna(data['Sales'].median(), inplace=True)
data['Quantity'].fillna(data['Quantity'].median(),
inplace=True)
data['Price'].fillna(data['Price'].median(), inplace=True)

# Fill missing categorical values with the mode
data['Store'].fillna(data['Store'].mode()[0], inplace=True)
data['Product'].fillna(data['Product'].mode()[0], inplace=True)
```

Converting Data Types

```
# Convert Date to datetime type
data['Date'] = pd.to_datetime(data['Date'])

# Check the data types
print(data.dtypes)
```

Removing Outliers

```
import numpy as np

# Remove outliers based on Z-score
from scipy.stats import zscore

data = data[(np.abs(zscore(data[['Sales', 'Quantity',
'Price']])) < 3).all(axis=1)]
```

Feature Engineering

```
# Extract month and year from the Date
data['Month'] = data['Date'].dt.month
data['Year'] = data['Date'].dt.year

# Drop the original Date column
data.drop(columns=['Date'], inplace=True)
```

3. Exploratory Data Analysis (EDA)

```
import matplotlib.pyplot as plt
import seaborn as sns

# Sales over time
plt.figure(figsize=(10, 6))
sns.lineplot(data=data, x='Month', y='Sales', hue='Year')
plt.title('Monthly Sales Over Years')
plt.show()

# Sales by store
plt.figure(figsize=(10, 6))
sns.barplot(data=data, x='Store', y='Sales')
plt.title('Sales by Store')
plt.show()

# Sales by product
```

```
plt.figure(figsize=(10, 6))
sns.barplot(data=data, x='Product', y='Sales')
plt.title('Sales by Product')
plt.show()
```

4. Machine Learning Modeling

Splitting the Data

```
from sklearn.model_selection import train_test_split

# Define features and target variable
X = data.drop(columns=['Sales'])
y = data['Sales']

# One-hot encoding for categorical variables
X = pd.get_dummies(X, drop_first=True)

# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)
```

Training a Simple Model

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Initialize the model
```

```
model = LinearRegression()

# Train the model
model.fit(X_train, y_train)

# Make predictions
y_pred = model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f'Mean Squared Error: {mse}')
print(f'R^2 Score: {r2}')
```

5. Model Interpretation and Conclusion

You can interpret the model by looking at the coefficients:

```
coefficients = pd.DataFrame(model.coef_, X.columns,
columns=['Coefficient'])
print(coefficients)
```

Summary

In this guide, we performed the following steps:

1. Loaded and cleaned the coffee sales data.
2. Conducted exploratory data analysis (EDA) to visualize sales trends.
3. Prepared the data for machine learning by handling categorical variables and splitting the dataset.
4. Trained a simple linear regression model to predict sales.
5. Evaluated the model's performance.

This is a basic example. For a more robust analysis, you might consider advanced techniques like cross-validation, feature selection, and trying different algorithms.

Sample code

Objective¶

This dataset contains detailed records of coffee sales from a vending machine. The dataset spans from March 2024 to Present time, capturing daily transaction data. In this notebook, we are going to use EDA to discover the customer's purchasing patterns and sales trends which can aid in the inventory planning.

Import packages

In [1]:

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt
import seaborn as sns
import datetime as dt
```

```
import warnings
warnings.filterwarnings('ignore')

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

```
/kaggle/input/coffee-sales/index.csv
```

Load data

In [2]:

```
coffee_data = pd.read_csv('/kaggle/input/coffee-sales/index.csv')
```

EDA

In [3]:

```
coffee_data.head()
```

Out[3]:

	date	datetime	cash_type	card	money	coffee_name
0	2024-03-01	2024-03-01 10:15:50.520	card	ANON-0000-0000-0001	38.7	Latte
1	2024-03-01	2024-03-01 12:19:22.539	card	ANON-0000-0000-0002	38.7	Hot Chocolate

2	2024-03-01	2024-03-01 12:20:18.089	card	ANON-0000-0000-0002	38.7	Hot Chocolate
3	2024-03-01	2024-03-01 13:46:33.006	card	ANON-0000-0000-0003	28.9	Americano
4	2024-03-01	2024-03-01 13:48:14.626	card	ANON-0000-0000-0004	38.7	Latte

In [4]:

```
coffee_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1133 entries, 0 to 1132
Data columns (total 6 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   date         1133 non-null   object 
 1   datetime     1133 non-null   object 
 2   cash_type    1133 non-null   object 
 3   card         1044 non-null   object 
 4   money        1133 non-null   float64
 5   coffee_name  1133 non-null   object 
dtypes: float64(1), object(5)
memory usage: 53.2+ KB
```

In [5]:

```
coffee_data.isnull().sum()
```

Out[5]:

```
date            0
datetime        0
cash_type       0
card           89
```

```
money          0
coffee_name    0

dtype: int64
```

In [6]:

```
coffee_data.duplicated().sum()
```

Out[6]:

```
0
```

In [7]:

```
coffee_data.describe().T
```

Out[7]:

	count	mean	std	min	25%	50%	75%	max
money	1133.0	33.105808	5.035366	18.12	28.9	32.82	37.72	40.0

In [8]:

```
coffee_data.loc[:,['cash_type','card','coffee_name']].describe().T
```

Out[8]:

	count	unique	top	freq

cash_type	1133	2	card	1044
card	1044	446	ANON-0000-0000-0012	88
coffee_name	1133	8	Americano with Milk	268

- There are 1033 transactions in the data.
- 89 missing values in the column 'card'.
- No duplicates.
- 2 unique values of 'cash_type'.
- 8 different coffee types with 'Americano with Milk' is the most popular product.

Let's check the transactions with missing value in 'card'.

In [9]:

```
coffee_data[coffee_data['card'].isnull()]['cash_type'].value_counts()
```

Out[9]:

```
cash_type
cash    89

Name: count, dtype: int64
```

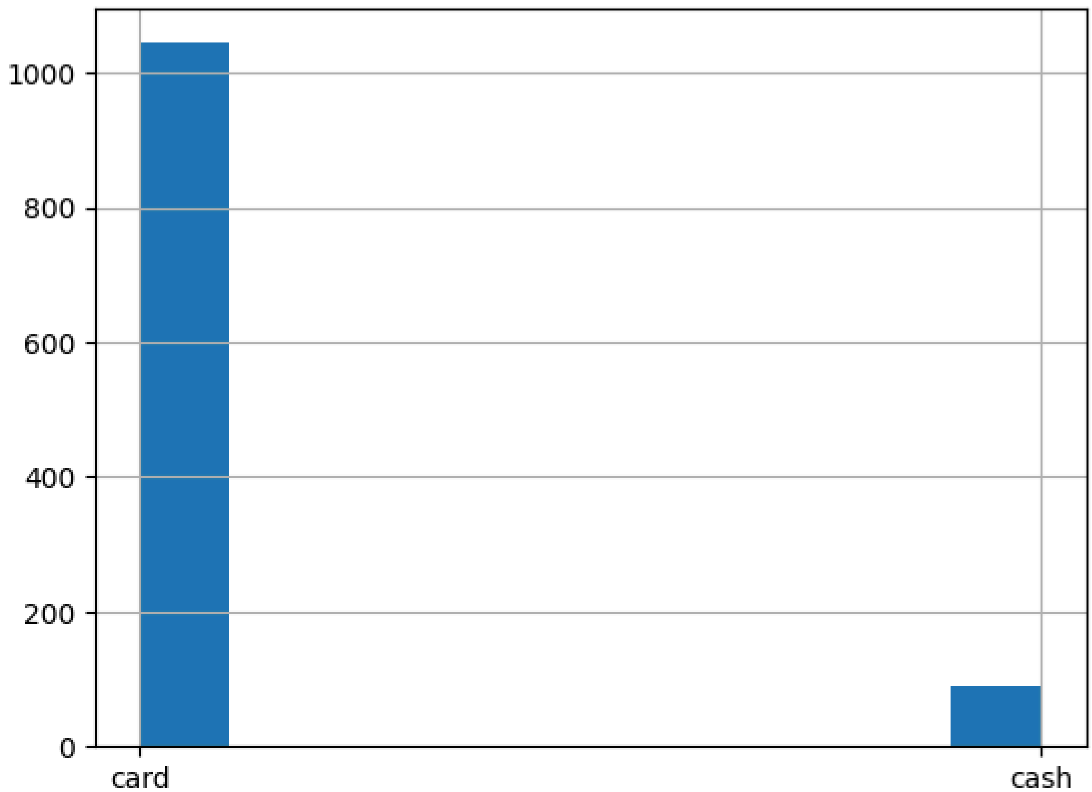
All of the transactions with null 'card' information are from cash users.

In [10]:

```
coffee_data['cash_type'].hist()
```

Out[10]:

```
<Axes: >
```



In [11]:

```
coffee_data['cash_type'].value_counts(normalize=True)
```

Out[11]:

```
cash_type
card    0.921447
cash    0.078553

Name: proportion, dtype: float64
```

~92% of the transactions are from card users.

In [12]:

```
pd.DataFrame(coffee_data['coffee_name'].value_counts(normalize=True).sort_values(asc
ending=False).round(4)*100)
```

Out[12]:

	proportion
coffee_name	
Americano with Milk	23.65
Latte	21.45
Cappuccino	17.30
Americano	14.92
Cortado	8.74
Hot Chocolate	6.53
Espresso	4.32
Cocoa	3.09

Americano with Milk and Latte are our most popular coffee products. In the second tier are Cappuccino and Americano, while Cortado, Hot Chocolate, Espresso, and Cocoa are less popular.

Next, let's conduct data transformations for further analysis.

In [13]:

```
#Convert date and datetime to datetme format
coffee_data['date']=pd.to_datetime(coffee_data['date'])
coffee_data['datetime']=pd.to_datetime(coffee_data['datetime'])
#Create column of Month, Weekdays, and Hours
coffee_data['month']=coffee_data['date'].dt.strftime('%Y-%m')
coffee_data['day']=coffee_data['date'].dt.strftime('%w')
coffee_data['hour']=coffee_data['datetime'].dt.strftime('%H')
```

In [14]:

```
coffee_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1133 entries, 0 to 1132
Data columns (total 9 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   date         1133 non-null   datetime64[ns]
 1   datetime     1133 non-null   datetime64[ns]
 2   cash_type    1133 non-null   object
 3   card         1044 non-null   object
 4   money        1133 non-null   float64
 5   coffee_name  1133 non-null   object
 6   month        1133 non-null   object
 7   day          1133 non-null   object
 8   hour         1133 non-null   object
dtypes: datetime64[ns](2), float64(1), object(6)
memory usage: 79.8+ KB
```

In [15]:

```
coffee_data.head()
```

Out[15]:

	date	datetime	cash_type	card	money	coffee_name	month	day	hour
0	2024-03-01	2024-03-01 10:15:50.520	card	ANON-0000-0000-0001	38.7	Latte	2024-03	5	10
1	2024-03-01	2024-03-01 12:19:22.539	card	ANON-0000-0000-0002	38.7	Hot Chocolate	2024-03	5	12
2	2024-03-01	2024-03-01 12:20:18.089	card	ANON-0000-0000-0002	38.7	Hot Chocolate	2024-03	5	12
3	2024-03-01	2024-03-01 13:46:33.006	card	ANON-0000-0000-0003	28.9	Americano	2024-03	5	13
4	2024-03-01	2024-03-01 13:48:14.626	card	ANON-0000-0000-0004	38.7	Latte	2024-03	5	13

In [16]:

```
[coffee_data['date'].min(),coffee_data['date'].max()]
```

Out[16]:

```
[Timestamp('2024-03-01 00:00:00'), Timestamp('2024-07-31 00:00:00')]
```

The time range of this data set is from 2023-3-1 to 2024-7-31

Let's first check the overal revenue by products.

In [17]:

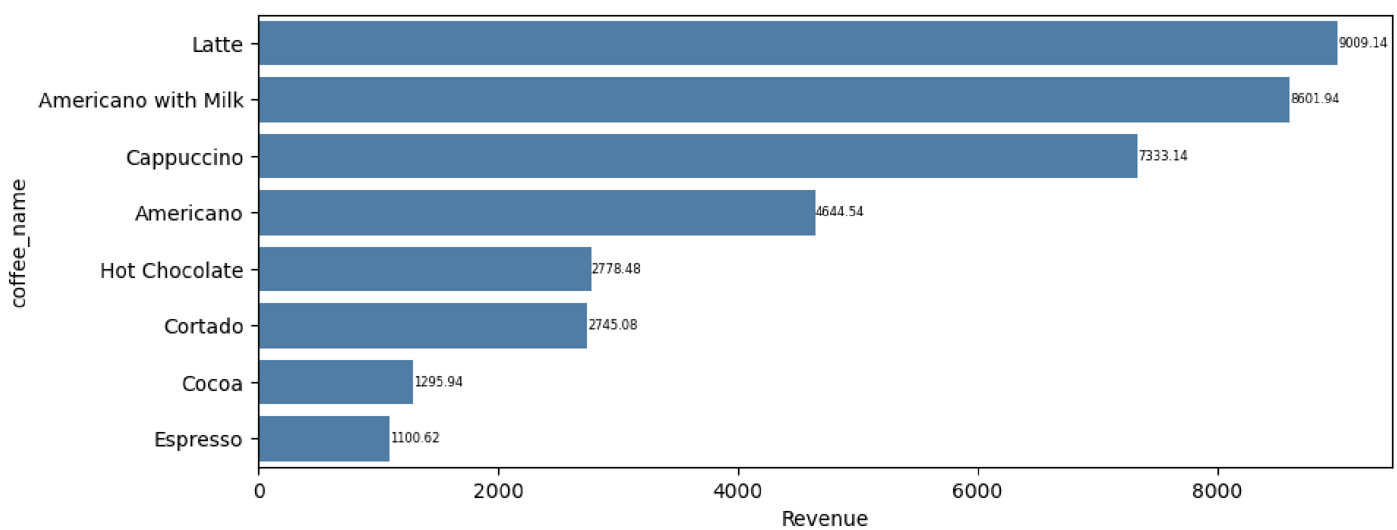
```
revenue_data =
coffee_data.groupby(['coffee_name']).sum(['money']).reset_index().sort_values(by='mo
ney',ascending=False)
```

In [18]:

```
plt.figure(figsize=(10,4))
ax = sns.barplot(data=revenue_data,x='money',y='coffee_name',color='steelblue')
ax.bar_label(ax.containers[0], fontsize=6)
plt.xlabel('Revenue')
```

Out[18]:

```
Text(0.5, 0, 'Revenue')
```



Latte is the product with the highest revenue, while Expresso is the one at the bottom. Then let's check the monthly data.

In [19]:

```
monthly_sales =
coffee_data.groupby(['coffee_name','month']).count()['date'].reset_index().rename(co
lumns={'date':'count'}).pivot(index='month',columns='coffee_name',values='count').re
set_index()
monthly_sales
```

Out[19]:

coffee_name	month	Americano	Americano with Milk	Cappuccino	Cocoa	Cortado	Espresso	Hot Chocolate	Latte
0	2024-03	36	34	20	6	30	10	22	48
1	2024-04	35	42	43	6	19	7	13	31
2	2024-05	48	58	55	9	17	8	14	58
3	2024-06	14	69	46	5	19	10	14	50
4	2024-07	36	65	32	9	14	14	11	56

In [20]:

```
monthly_sales.describe().T.loc[:,['min','max']]
```

Out[20]:

	min	max

coffee_name		
Americano	14.0	48.0
Americano with Milk	34.0	69.0
Cappuccino	20.0	55.0
Cocoa	5.0	9.0
Cortado	14.0	30.0
Espresso	7.0	14.0
Hot Chocolate	11.0	22.0
Latte	31.0	58.0

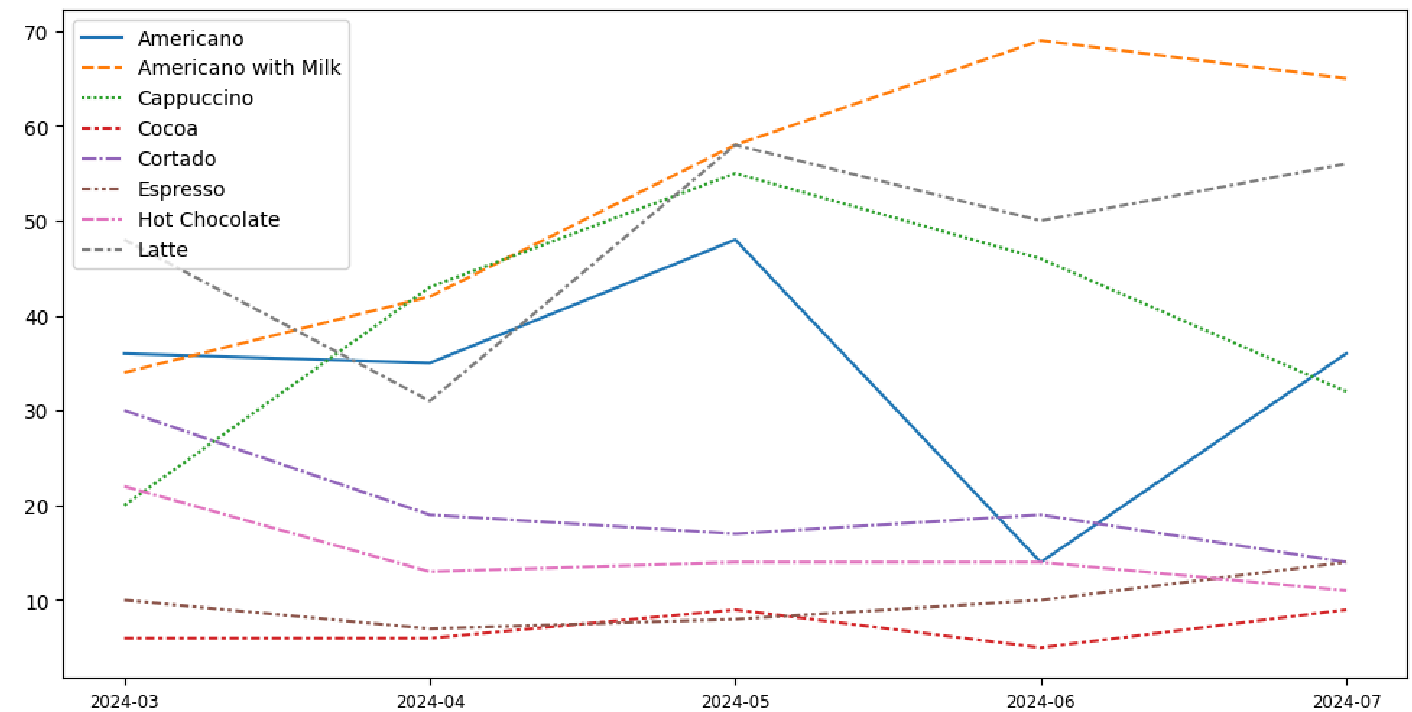
In [21]:

```
plt.figure(figsize=(12,6))
sns.lineplot(data=monthly_sales)
plt.legend(loc='upper left')
plt.xticks(range(len(monthly_sales['month'])),monthly_sales['month'],size='small')
```

Out[21]:

```
([<matplotlib.axis.XTick at 0x7d45ae8a0430>,
  <matplotlib.axis.XTick at 0x7d45ae8a0400>,
  <matplotlib.axis.XTick at 0x7d45ae8a2ef0>,
  <matplotlib.axis.XTick at 0x7d45ae8d3ee0>,
  <matplotlib.axis.XTick at 0x7d45ae9149d0>],
 [Text(0, 0, '2024-03'),
  Text(1, 0, '2024-04'),
  Text(2, 0, '2024-05'),
  Text(3, 0, '2024-06'),

  Text(4, 0, '2024-07')])
```



As shown in the line chart above, Americano with Milk and Latte, and Cappuccino are top selling coffee types, while Cocoa and Expresso have lowest sales. Additionally, Americano with Milk and Latte show an upward trending.

In [22]:

```
weekday_sales =
coffee_data.groupby(['day']).count()['date'].reset_index().rename(columns={'date':'c
ount'})
weekday_sales
```

Out[22]:

	day	count
0	0	151
1	1	151
2	2	185
3	3	165
4	4	164
5	5	163
6	6	154

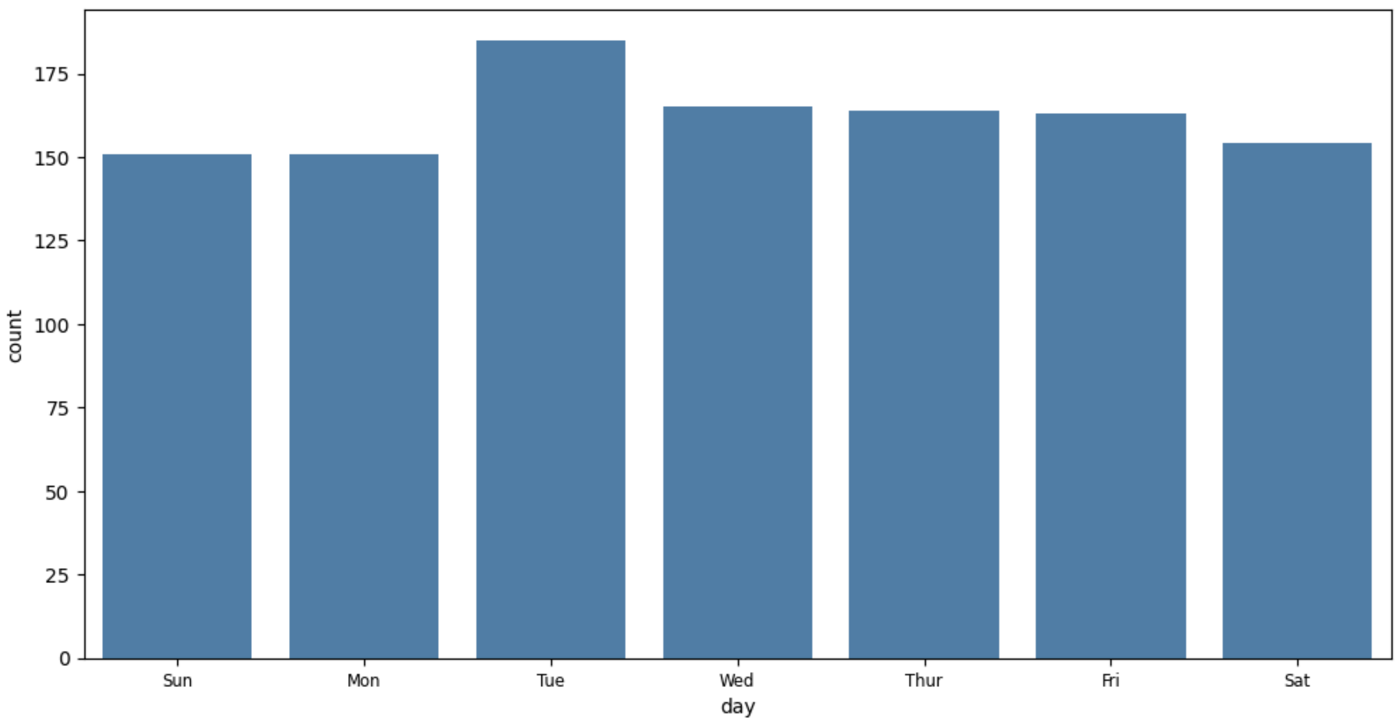
In [23]:

```
plt.figure(figsize=(12,6))
sns.barplot(data=weekday_sales,x='day',y='count',color='steelblue')
plt.xticks(range(len(weekday_sales['day'])),['Sun','Mon','Tue','Wed','Thur','Fri','Sat'],size='small')
```

Out[23]:

```
([<matplotlib.axis.XTick at 0x7d45aea5b070>,
  <matplotlib.axis.XTick at 0x7d45aea5b040>,
  <matplotlib.axis.XTick at 0x7d45aea5af50>,
  <matplotlib.axis.XTick at 0x7d45aeaa1240>,
  <matplotlib.axis.XTick at 0x7d45aeaa1cf0>,
  <matplotlib.axis.XTick at 0x7d45cf8c5f00>,
  <matplotlib.axis.XTick at 0x7d45aeaa29b0>],
 [Text(0, 0, 'Sun'),
  Text(1, 0, 'Mon'),
  Text(2, 0, 'Tue'),
  Text(3, 0, 'Wed'),
  Text(4, 0, 'Thur'),
  Text(5, 0, 'Fri'),

  Text(6, 0, 'Sat')])
```



The bar chart reveals that Tuesday has the highest sales of the week, while sales on the other days are relatively similar.

In [24]:

```
daily_sales =
coffee_data.groupby(['coffee_name','date']).count()['datetime'].reset_index().reset_
```

```python
index().rename(columns={'datetime':'count'}).pivot(index='date',columns='coffee_name
',values='count').reset_index().fillna(0)
daily_sales
```

Out[24]:

coffee_name	date	Americano	Americano with Milk	Cappuccino	Cocoa	Cortado	Espresso	Hot Chocolate	Latte
0	2024-03-01	1.0	4.0	0.0	1.0	0.0	0.0	3.0	2.0
1	2024-03-02	3.0	3.0	0.0	0.0	0.0	0.0	0.0	1.0
2	2024-03-03	1.0	2.0	0.0	1.0	2.0	0.0	2.0	2.0
3	2024-03-04	0.0	1.0	0.0	0.0	0.0	1.0	0.0	2.0
4	2024-03-05	0.0	0.0	0.0	1.0	1.0	0.0	4.0	3.0
...	...	...	...	...	...	...	...	...	...
145	2024-07-27	0.0	5.0	4.0	0.0	0.0	2.0	0.0	2.0

146	2024-07-28	0.0	1.0	0.0	0.0	0.0	1.0	0.0	1.0
147	2024-07-29	3.0	2.0	2.0	1.0	0.0	0.0	2.0	1.0
148	2024-07-30	2.0	12.0	2.0	0.0	3.0	2.0	0.0	3.0
149	2024-07-31	2.0	6.0	1.0	2.0	4.0	0.0	0.0	7.0

150 rows × 9 columns

In [25]:

```
daily_sales.iloc[:,1:].describe().T.loc[:,['min','max']]
```

Out[25]:

	min	max
coffee_name		
Americano	0.0	5.0

Americano with Milk	0.0	12.0
Cappuccino	0.0	9.0
Cocoa	0.0	2.0
Cortado	0.0	4.0
Espresso	0.0	4.0
Hot Chocolate	0.0	4.0
Latte	0.0	7.0

This table provides us the infomation of how many of each products can be sold in each day.

In [26]:

```
hourly_sales =
coffee_data.groupby(['hour']).count()['date'].reset_index().rename(columns={'date':'
count'})
hourly_sales
```

Out[26]:

	hour	count
0	07	13
1	08	44
2	09	50
3	10	133
4	11	103
5	12	87
6	13	78
7	14	76
8	15	65

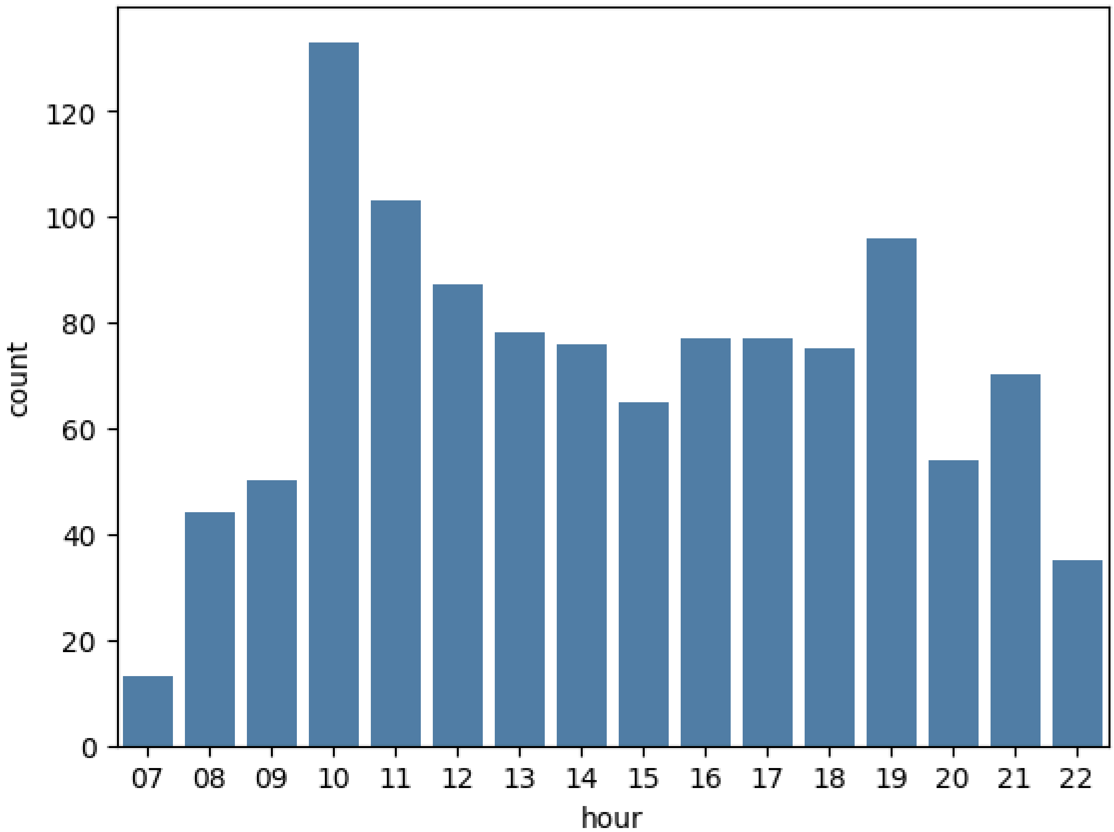
9	16	77
10	17	77
11	18	75
12	19	96
13	20	54
14	21	70
15	22	35

In [27]:

```
sns.barplot(data=hourly_sales,x='hour',y='count',color='steelblue')
```

Out[27]:

```
<Axes: xlabel='hour', ylabel='count'>
```



Overall, two peak hours within each day can be observed: 10:00am and 7:00pm. Then, let's check if any difference for different products.

In [28]:

```
hourly_sales_by_coffee =
coffee_data.groupby(['hour','coffee_name']).count()['date'].reset_index().rename(col
umns={'date':'count'}).pivot(index='hour',columns='coffee_name',values='count').fill
na(0).reset_index()
hourly_sales_by_coffee
```

Out[28]:

coffee_name	hour	Americano	Americano with Milk	Cappuccino	Cocoa	Cortado	Espresso	Hot Chocolate	Latte

0	07	5.0	4.0	1.0	0.0	1.0	0.0	0.0	2.0
1	08	10.0	7.0	8.0	1.0	6.0	0.0	0.0	12.0
2	09	8.0	16.0	6.0	1.0	5.0	3.0	0.0	11.0
3	10	20.0	31.0	10.0	4.0	8.0	2.0	7.0	51.0
4	11	21.0	25.0	16.0	1.0	13.0	6.0	8.0	13.0
5	12	14.0	26.0	15.0	3.0	7.0	6.0	3.0	13.0
6	13	18.0	18.0	10.0	2.0	12.0	3.0	4.0	11.0
7	14	15.0	18.0	13.0	4.0	6.0	5.0	2.0	13.0
8	15	14.0	15.0	8.0	0.0	3.0	4.0	6.0	15.0
9	16	10.0	18.0	12.0	3.0	12.0	5.0	4.0	13.0
10	17	9.0	11.0	18.0	4.0	6.0	4.0	7.0	18.0

11	18	9.0	16.0	12.0	2.0	5.0	5.0	10.0	16.0
12	19	5.0	18.0	34.0	2.0	5.0	1.0	9.0	22.0
13	20	1.0	12.0	13.0	6.0	5.0	3.0	6.0	8.0
14	21	5.0	25.0	13.0	1.0	3.0	1.0	3.0	19.0
15	22	5.0	8.0	7.0	1.0	2.0	1.0	5.0	6.0

In [29]:

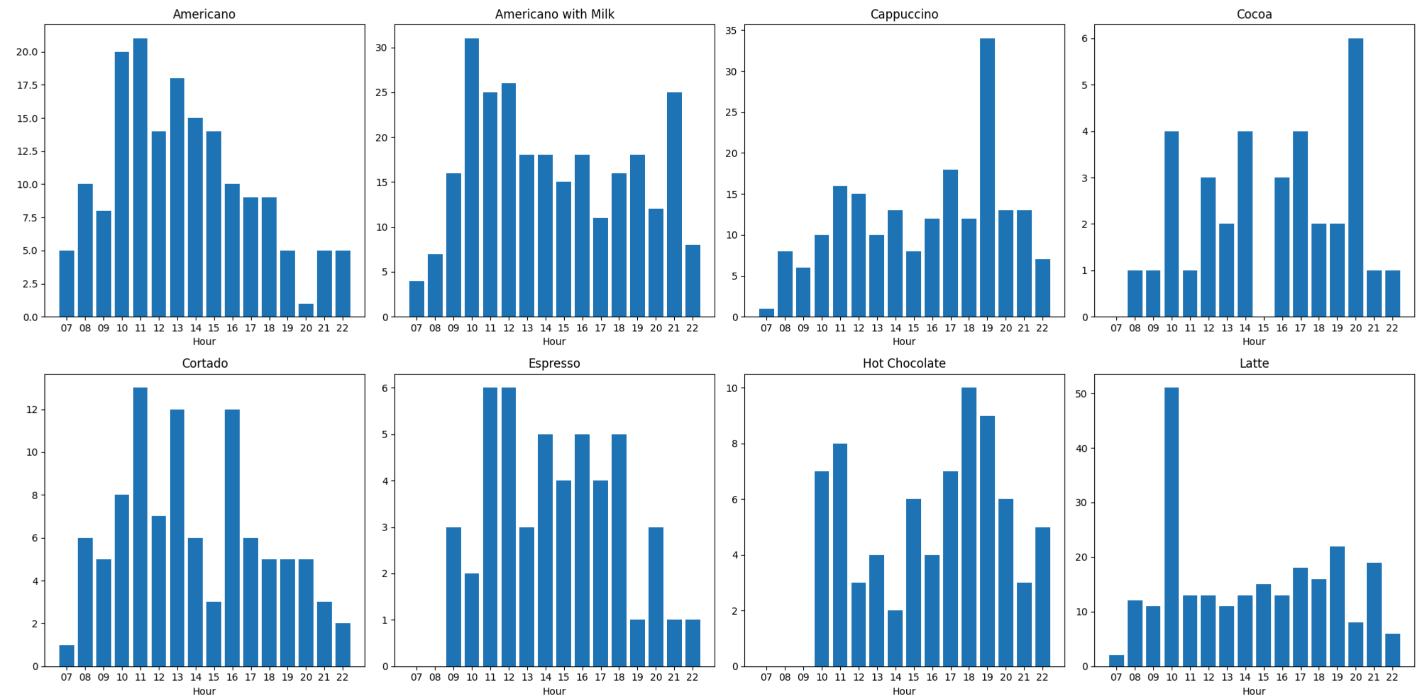
```
fig, axs = plt.subplots(2, 4, figsize=(20, 10))

# Flatten the array of subplots for easy iteration
axs = axs.flatten()

# Loop through each column in the DataFrame, skipping the 'Index' column
for i, column in enumerate(hourly_sales_by_coffee.columns[1:]):  # Skip the first
column ('Index')
    axs[i].bar(hourly_sales_by_coffee['hour'], hourly_sales_by_coffee[column])
    axs[i].set_title(f'{column}')
    axs[i].set_xlabel('Hour')
    #axs[i].set_ylabel('Sales')

plt.tight_layout()

# Show the plot
plt.show()
```



The plots above illustrate the shopping traffic for each product throughout the day. Notably, all products experience a peak in traffic around 10:00 AM, with this trend being particularly pronounced for Latte. Additionally, Cappuccino, Cocoa, and Hot Chocolate tend to be more popular during the evening hours, specifically between 6:00pm and 8:00pm.

Conclusion

From the analysis above, we have uncovered valuable insights into customer shopping patterns on a daily and weekly basis. We have identified the most popular coffee products and observed the shopping trends over time. These findings are instrumental in optimizing inventory planning, designing the layout of vending machines, and determining the ideal restock times for coffee products.

1 Reference link

2 Reference link for ML project