```
import numpy as np       #NUMPY IS USED FOR NUMERICAL OPERATIONS
import pandas as pd # PANDAS FOR IMPORTING THE DATASET
from sklearn.model_selection import train_test_split    #TO SPLIT DATA INTO TRAINING AND TESTING DATA
from  sklearn.tree import DecisionTreeClassifier    #FOR DECSION TREE CLASSIFIER
from sklearn.metrics import accuracy_score   #TO CHECK ACCURACY
import matplotlib.pyplot as plt     # FOR DATA VISUALIZATION PURPOSE
from sklearn import tree   # TO VISUALIZE THE TREE
import seaborn as sns     # FOR DATA VISUALIZATION

from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report, confusion_matrix
```

LOADING THE DATASET:

```
df = pd.read_csv("/content/index.csv")    # using the Pandas library's read_csv function. The dataset is st
df
```

	date	datetime	cash_type	card	money	coffee_name
0	2024-03-01	2024-03-01 10:15:50.520	card	ANON-0000-0000-0001	38.70	Latte
1	2024-03-01	2024-03-01 12:19:22.539	card	ANON-0000-0000-0002	38.70	Hot Chocolate
2	2024-03-01	2024-03-01 12:20:18.089	card	ANON-0000-0000-0002	38.70	Hot Chocolate
3	2024-03-01	2024-03-01 13:46:33.006	card	ANON-0000-0000-0003	28.90	Americano
4	2024-03-01	2024-03-01 13:48:14.626	card	ANON-0000-0000-0004	38.70	Latte
...	...	...	...	...	...	...
1128	2024-07-31	2024-07-31 20:53:35.077	card	ANON-0000-0000-0443	23.02	Cortado
1129	2024-07-31	2024-07-31 20:59:25.013	card	ANON-0000-0000-0040	27.92	Americano with Milk
1130	2024-07-31	2024-07-31 21:26:26.000	card	ANON-0000-0000-0444	32.82	Latte
1131	2024-07-31	2024-07-31 21:54:11.824	card	ANON-0000-0000-0445	32.82	Latte
1132	2024-07-31	2024-07-31 21:55:16.570	card	ANON-0000-0000-0446	32.82	Latte

1133 rows × 6 columns

```
df.head(10)
```

	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
5	2024-03-01	2024-03-01 15:39:47.726	card	ANON-0000-0000-0005	33.8	Americano with Milk
6	2024-03-01	2024-03-01 16:19:02.756	card	ANON-0000-0000-0006	38.7	Hot Chocolate
7	2024-03-01	2024-03-01 18:39:03.580	card	ANON-0000-0000-0007	33.8	Americano with Milk
8	2024-03-01	2024-03-01 19:22:01.762	card	ANON-0000-0000-0008	38.7	Cocoa
9	2024-03-01	2024-03-01 19:23:15.887	card	ANON-0000-0000-0008	33.8	Americano with Milk

```
df.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
```

```
df.shape
```

```
(1133, 6)
```

```
df.describe()
```

	money
count	1133.000000
mean	33.105808
std	5.035366
min	18.120000
25%	28.900000
50%	32.820000
75%	37.720000
max	40.000000

```
df.isnull().sum()
```

	0
date	0
datetime	0
cash_type	0
card	89
money	0
coffee_name	0

```
df.columns
```

```
Index(['date', 'datetime', 'cash_type', 'card', 'money', 'coffee_name'], dtype='object')
```

```
df.dropna(inplace=True)
```

```
df.isnull().sum()
```

	0
date	0
datetime	0
cash_type	0
card	0
money	0
coffee_name	0

```
df['date']
```

	date
0	2024-03-01
1	2024-03-01
2	2024-03-01
3	2024-03-01
4	2024-03-01
...	...
1128	2024-07-31
1129	2024-07-31
1130	2024-07-31
1131	2024-07-31
1132	2024-07-31

1044 rows × 1 columns

```
df.dtypes
```

	0
date	object
datetime	object
cash_type	object
card	object
money	float64
coffee_name	object

```
df['date'] = pd.to_datetime(df['date'])
```

```
df.dtypes
```

	0
date	datetime64[ns]
datetime	object
cash_type	object
card	object
money	float64
coffee_name	object

```
df.head()
```

	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

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

	count	unique	top	freq
cash_type	1044	1	card	1044
card	1044	446	ANON-0000-0000-0012	88
coffee_name	1044	8	Americano with Milk	253

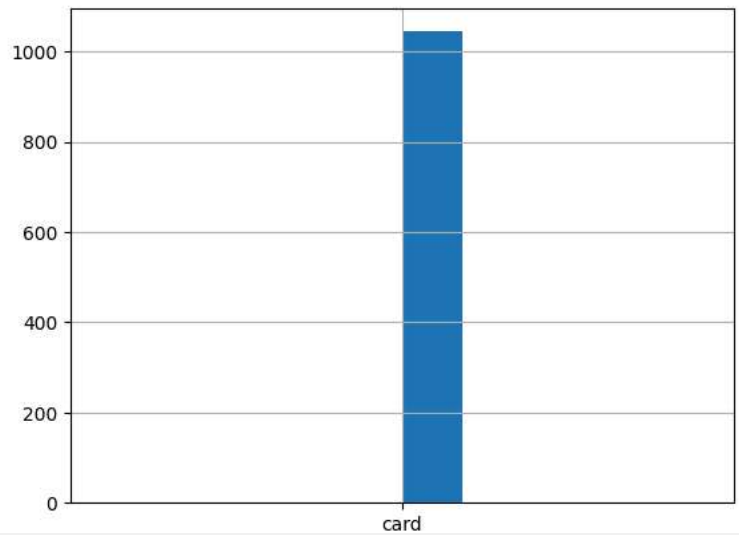
```
Start coding or generate with AI.
```

- There are 1044 transactions in the data.
- 1 unique value of 'cash_type' and that is -> 'card'
- Americano with milk is the most popular product

```
Start coding or generate with AI.
```

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

```
<Axes: >
```




```
Start coding or generate with AI.
```

Almost all transactions are made from card payment

```
Start coding or generate with AI.
```

```
df['coffee_name'].value_counts(normalize=True).sort_values(ascending=False).round(4)*100
```

coffee_name	proportion
Americano with Milk	24.23
Latte	20.88
Cappuccino	17.34
Americano	14.85
Cortado	9.00
Hot Chocolate	6.51
Espresso	4.21
Cocoa	2.97

- In this we can see the most popular products which are arranged in descending order
- Americano with Milk is at the top where as the second place belongs to Latte followed by other coffees

```
Start coding or generate with AI.
```

```
 #Convert date and datetime to datetme format
df['date']=pd.to_datetime(df['date'])
df['datetime']=pd.to_datetime(df['datetime'])
```

```
#Create column of Month, Weekdays, and Hours
df['month']=df['date'].dt.strftime('%Y-%m')
df['day']=df['date'].dt.strftime('%w')
df['hour']=df['datetime'].dt.strftime('%H')
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1044 entries, 0 to 1132
Data columns (total 9 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   date         1044 non-null   datetime64[ns]
 1   datetime     1044 non-null   datetime64[ns]
 2   cash_type    1044 non-null   object
 3   card         1044 non-null   object
 4   money        1044 non-null   float64
 5   coffee_name  1044 non-null   object
 6   month        1044 non-null   object
 7   day          1044 non-null   object
 8   hour         1044 non-null   object
dtypes: datetime64[ns](2), float64(1), object(6)
memory usage: 113.9+ KB
```

```
df.head()
```

	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

```
Start coding or generate with AI.
```

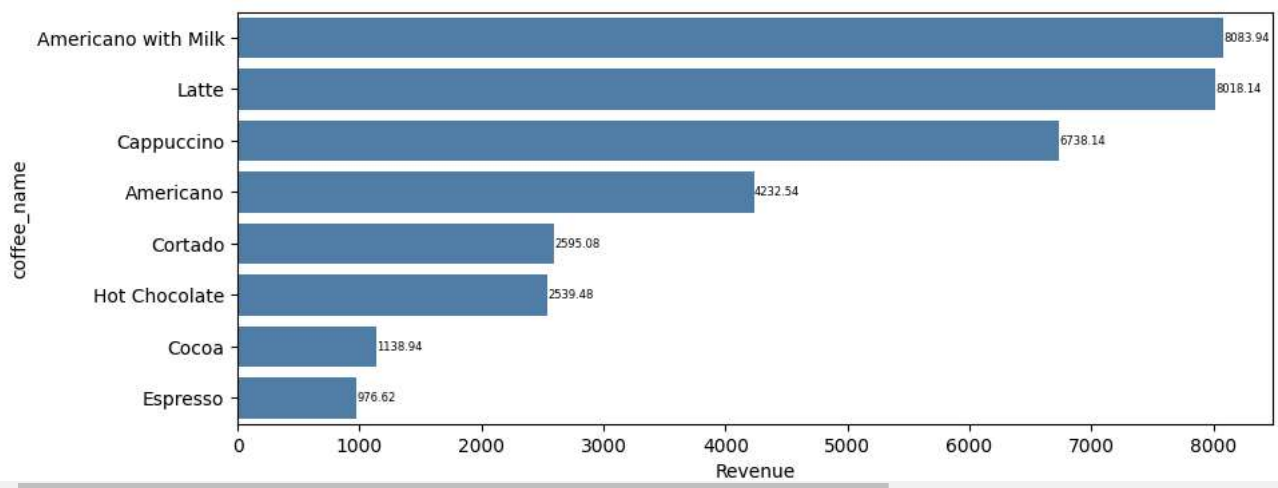
```
#  Let's first check the overal revenue by products.
```

```
revenue = df.groupby(['coffee_name']).sum(['money']).sort_values(by='money', ascending=False)
revenue
```

coffee_name	money
Americano with Milk	8083.94
Latte	8018.14
Cappuccino	6738.14
Americano	4232.54
Cortado	2595.08
Hot Chocolate	2539.48
Cocoa	1138.94
Espresso	976.62

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

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



The product with the highest revenue is Americano with Milk whereas Espresso is at the bottom

```
Start coding or generate with AI.
```

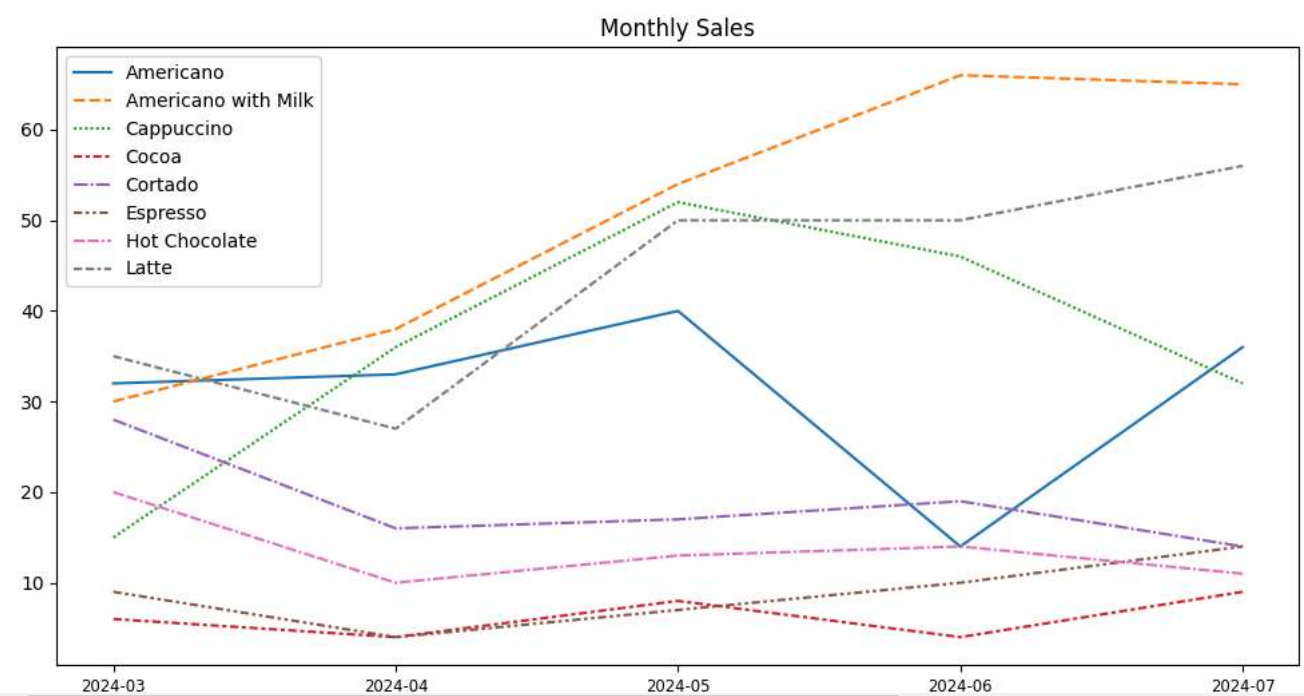
```
monthly_sales = df.groupby(['coffee_name','month']).count()['date'].reset_index().rename(columns={'date':'
monthly_sales
```

	coffee_name	month	Americano	Americano with Milk	Cappuccino	Cocoa	Cortado	Espresso	Hot Chocolate	Latte
0		2024-03	32	30	15	6	28	9	20	35
1		2024-04	33	38	36	4	16	4	10	27
2		2024-05	40	54	52	8	17	7	13	50
3		2024-06	14	66	46	4	19	10	14	50
4		2024-07	36	65	32	9	14	14	11	56

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

coffee_name	min	max
Americano	14.0	40.0
Americano with Milk	30.0	66.0
Cappuccino	15.0	52.0
Cocoa	4.0	9.0
Cortado	14.0	28.0
Espresso	4.0	14.0
Hot Chocolate	10.0	20.0
Latte	27.0	56.0

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



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.

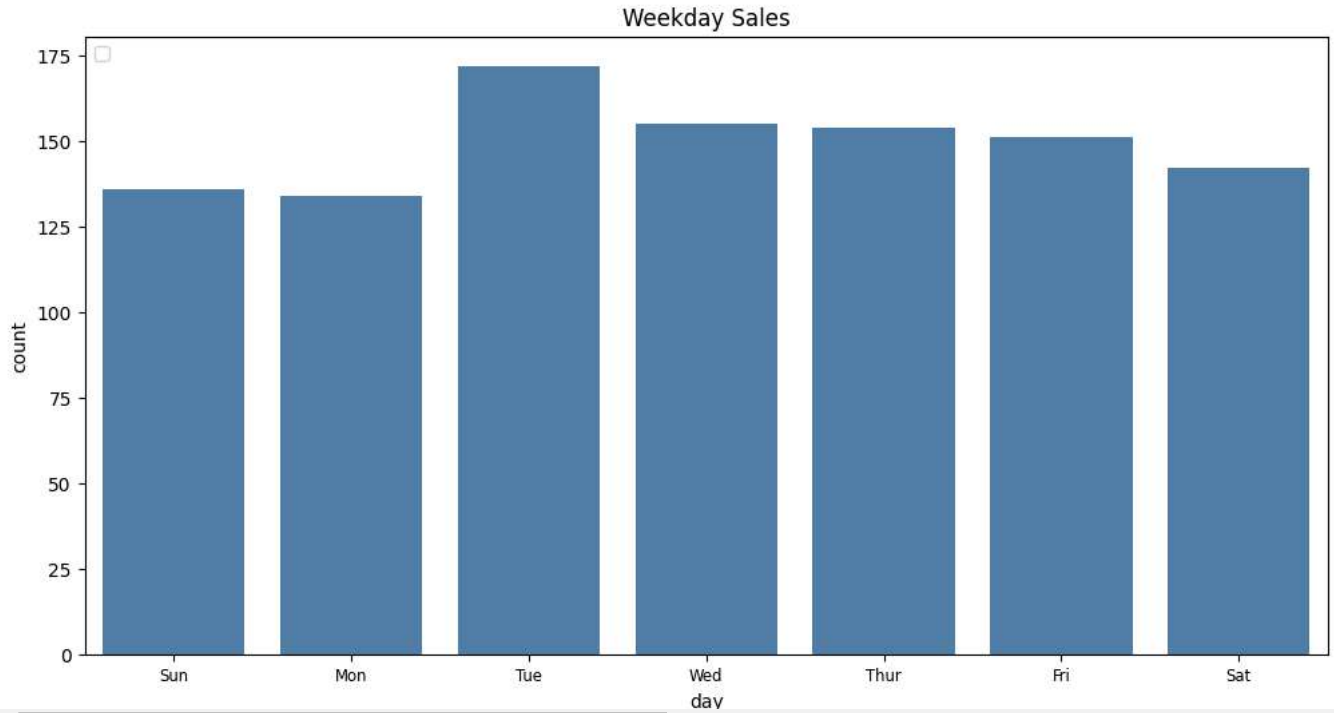
```
Start coding or generate with AI.
```

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

	day	count
0	0	136
1	1	134
2	2	172
3	3	155
4	4	154
5	5	151
6	6	142

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

```
WARNING:matplotlib.legend:No artists with labels found to put in legend.  Note that artists whose label start with an underscore are igr
```



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

```
Start coding or generate with AI.
```

```
daily_sales = df.groupby(['coffee_name','date']).count()['datetime'].reset_index().reset_index().rename(cc
daily_sales
```

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	0.0
2	2024-03-03	1.0	2.0	0.0	1.0	2.0	0.0	2.0	1.0
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

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

coffee_name	min	max
Americano	0.0	4.0
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	3.0
Hot Chocolate	0.0	4.0
Latte	0.0	7.0

```
Start coding or generate with AI.
```

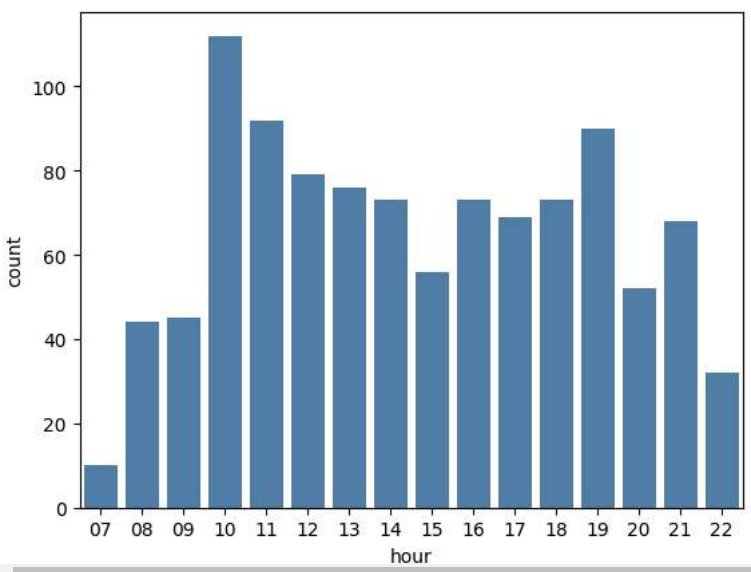
```
hourly_sales =  df.groupby(['hour']).count()['date'].reset_index().rename(columns={'date':'count'})
hourly_sales
```

	hour	count
0	07	10
1	08	44
2	09	45
3	10	112
4	11	92
5	12	79
6	13	76
7	14	73
8	15	56
9	16	73
10	17	69
11	18	73
12	19	90
13	20	52
14	21	68
15	22	32

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

```

```
<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.

```
Start coding or generate with AI.
```

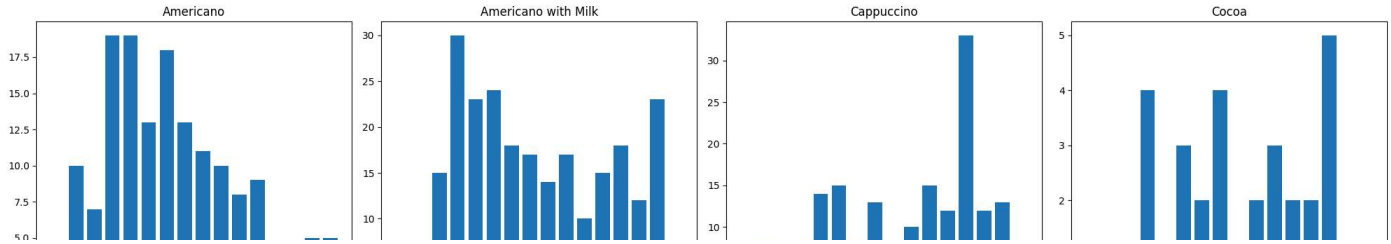
```
hourly_sales_by_coffee = df.groupby(['hour','coffee_name']).count()['date'].reset_index().rename(columns={
hourly_sales_by_coffee
```

coffee_name	hour	Americano	Americano with Milk	Cappuccino	Cocoa	Cortado	Espresso	Hot Chocolate	Latte
0	07	3.0	3.0	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	7.0	15.0	6.0	0.0	5.0	2.0	0.0	10.0
3	10	19.0	30.0	8.0	4.0	7.0	1.0	7.0	36.0
4	11	19.0	23.0	14.0	1.0	12.0	6.0	7.0	10.0
5	12	13.0	24.0	15.0	3.0	5.0	4.0	3.0	12.0
6	13	18.0	18.0	8.0	2.0	12.0	3.0	4.0	11.0
7	14	13.0	17.0	13.0	4.0	6.0	5.0	2.0	13.0
8	15	11.0	14.0	7.0	0.0	2.0	4.0	4.0	14.0
9	16	10.0	17.0	10.0	2.0	12.0	5.0	4.0	13.0
10	17	8.0	10.0	15.0	3.0	6.0	4.0	6.0	17.0
11	18	9.0	15.0	12.0	2.0	5.0	5.0	10.0	15.0
12	19	4.0	18.0	33.0	2.0	5.0	0.0	8.0	20.0
13	20	1.0	12.0	12.0	5.0	5.0	3.0	6.0	8.0
14	21	5.0	23.0	13.0	1.0	3.0	1.0	3.0	19.0
15	22	5.0	7.0	6.0	1.0	2.0	1.0	4.0	6.0

```
Start coding or generate with AI.
```

```
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:]):
  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()
```


Americano
Americano with Milk
Cappuccino
Cocoa