Supermarket_notebook_final_edition

July 10, 2023

1 Final Project

Welcome to the final practical project for our course on Data Science Bootcamp. Throughout this project, you will go through the entire data science process, starting from data loading and cleaning, all the way to running a model and making predictions. This hands-on project will provide you with valuable experience and allow you to apply the concepts and techniques you've learned in the course. Get ready to dive into real-world data analysis and build your skills as a data scientist!

1.1 Important Remarks:

- The ultimate goal of this project is to conduct comprehensive data analysis and build 2 models using the provided datasets.
- Code is not the only thing graded here. Well-written and understandable documentation of your code is to be expected
- Clear reasoning behind your choices in every step of the notebook is important. Be it the choice of a data cleaning technique or selecting certain features in your analysis or the choice of your 2 models.

2 Importing packages

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns #package for visualization
from sklearn.linear_model import LogisticRegression #package for a logistic␣
↪regression model
from sklearn.model_selection import train_test_split #split data
from sklearn.preprocessing import StandardScaler #Scaling data
from sklearn.neighbors import KNeighborsClassifier #package for KNN model
from sklearn.metrics import mean_absolute_error #calculate mean absolut error
from sklearn import metrics
from sklearn.metrics import accuracy_score #calculate accuracy score
```

3 Load the dataset into data

[2]:
```
df = pd.read_csv('supermarket_survey.csv', delimiter =";") #Load the␣
↪supermarket survey dataset into a pandas dataframe
```

4 Dataset overview and statistical summary

[3]:
```
df.info() #overview of columns, their data type and the number of non-null␣
↪entries
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 353 entries, 0 to 352
Data columns (total 46 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   randomInt              353 non-null    int64
 1   age                    345 non-null    object
 2   gender                 347 non-null    object
 3   district               334 non-null    object
 4   modeOfTransportation   341 non-null    object
 5   distance               338 non-null    object
 6   G03Q13amountOfPeople   345 non-null    object
 7   income                 331 non-null    float64
 8   frequency              339 non-null    object
 9   days[1]                353 non-null    object
 10  days[2]                353 non-null    object
 11  days[3]                353 non-null    object
 12  days[4]                353 non-null    object
 13  days[5]                353 non-null    object
 14  days[6]                353 non-null    object
 15  days[7]                353 non-null    object
 16  time[1]                353 non-null    object
 17  time[2]                353 non-null    object
 18  time[3]                353 non-null    object
 19  time[4]                353 non-null    object
 20  time[5]                353 non-null    object
 21  moneySpent             338 non-null    object
 22  orderingItems          334 non-null    object
 23  deliveringItems        333 non-null    object
 24  willingPayDelivery     166 non-null    object
 25  findProducts           334 non-null    object
 26  usingDiscounts         326 non-null    object
 27  preferCash             331 non-null    object
 28  preferCashless         329 non-null    object
 29  isRelaxing             327 non-null    object
 30  satisGeneralStore      332 non-null    float64
 31  satisMusic             288 non-null    float64
```

```
 32  satisQualityProducts     329 non-null    float64
 33  satisGeneralAssortment   330 non-null    float64
 34  satisVeganProducts       274 non-null    float64
 35  satisOrganicProducts     301 non-null    float64
 36  satisGlutenfreeProducts  209 non-null    float64
 37  satisAnimalProducts      307 non-null    float64
 38  ideasExtendedBusiness    324 non-null    float64
 39  ideasHelpCarry           322 non-null    float64
 40  ideasCustomerCouncil     318 non-null    float64
 41  ideasFreeWifi            324 non-null    float64
 42  ideasTouchDisplay        320 non-null    float64
 43  ideasSelfCheckout        323 non-null    float64
 44  ideasBikeParking         312 non-null    float64
 45  ideasUndergroundParking  300 non-null    float64
dtypes: float64(17), int64(1), object(28)
memory usage: 127.0+ KB
```

[4]:
```
#We can see that in almost all columns data is missing, but only a small␣
↪percentage. Only in columns like satisfaction with
#glutenfree products that are only relevant for a few people, more data is␣
↪missing. Only if we want to analyse these specific
#aspects handling the missing data is needed
```

[5]:
```
df.head(10) #returns the first 10 rows of the dataset -> inside in the data
```

[5]:
```
   randomInt    age             gender    district modeOfTransportation  \
0          4    NaN               Male      Godham              Own Car
1          4    NaN                NaN         NaN                  NaN
2          3  20-25             Female  Springtown              Own Car
3          4    NaN                NaN         NaN                  NaN
4          3  15-20               Male   Piltunder              Own Car
5          3  20-25  Prefer not to say  Metrapalis              Walking
6          2  60-65               Male      Godham              Own Car
7          1  40-45             Female      Godham              Bicycle
8          1  25-30               Male  Metrapalis              Walking
9          3  20-25             Female  Springtown              Bicycle

             distance G03Q13amountOfPeople    income    frequency days[1]  …  \
0              1-2km                    3  120000.0        Twice      No  …
1                NaN                  NaN       NaN          NaN      No  …
2               >7km                    2      15.0  Three times      No  …
3                NaN                  NaN    1337.0          NaN      No  …
4              1-2km                    4  250000.0        Twice      No  …
5  500 meters to 1km                    1     500.0        Twice      No  …
6              1-2km                    2    5000.0         Once      No  …
7  500 meters to 1km                    1       NaN        Twice      No  …
8  500 meters to 1km                    1     600.0   Four times     Yes  …
```

```
9  500 meters to 1km                    1    1200.0   Four times     Yes  …

  satisGlutenfreeProducts satisAnimalProducts ideasExtendedBusiness  \
0                     8.0                 7.0                   2.0
1                     NaN                 NaN                   NaN
2                     7.0                 NaN                   7.0
3                     NaN                 NaN                   NaN
4                     8.0                 1.0                   9.0
5                    10.0                10.0                   9.0
6                     NaN                 7.0                   5.0
7                     NaN                 7.0                   8.0
8                     6.0                 8.0                  10.0
9                     3.0                 7.0                  10.0

  ideasHelpCarry ideasCustomerCouncil ideasFreeWifi ideasTouchDisplay  \
0            4.0                  3.0           4.0               NaN
1            NaN                  NaN           NaN               NaN
2            7.0                  7.0           7.0               NaN
3            NaN                  NaN           NaN               NaN
4            2.0                  1.0          10.0              10.0
5            1.0                  1.0           9.0               9.0
6            2.0                  2.0           6.0               3.0
7            3.0                  6.0          10.0               8.0
8            2.0                  3.0           4.0               3.0
9            1.0                  2.0           5.0               2.0

  ideasSelfCheckout ideasBikeParking ideasUndergroundParking
0               4.0              NaN                     NaN
1               NaN              NaN                     NaN
2               7.0              7.0                     7.0
3               NaN              NaN                     NaN
4              10.0              8.0                     NaN
5              10.0              1.0                     1.0
6               9.0              9.0                     9.0
7              10.0              9.0                     1.0
8              10.0             10.0                     2.0
9              10.0             10.0                     3.0

[10 rows x 46 columns]
```

[6]:
```
#Due to the design of the survey a lot of columns have categorical data
```

[7]:
```
df.describe() #gives quantitative analysis about each row -> not all are helpful
```

[7]:
```
         randomInt         income  satisGeneralStore  satisMusic  \
count  353.000000     331.000000         332.000000  288.000000
mean     2.609065   66275.568882           7.424699    5.236111
```

std	1.105322	132542.950482	1.705790	2.507094
min	1.000000	-99932.000000	1.000000	1.000000
25%	2.000000	2290.000000	7.000000	3.000000
50%	3.000000	21000.000000	8.000000	5.000000
75%	4.000000	80284.000000	8.000000	7.000000
max	4.000000	999999.000000	10.000000	10.000000

	satisQualityProducts	satisGeneralAssortment	satisVeganProducts	\
count	329.000000	330.000000	274.000000	
mean	7.498480	7.278788	6.350365	
std	1.479792	1.674366	2.177444	
min	1.000000	1.000000	1.000000	
25%	7.000000	7.000000	5.000000	
50%	8.000000	8.000000	7.000000	
75%	8.000000	8.000000	8.000000	
max	10.000000	10.000000	10.000000	

	satisOrganicProducts	satisGlutenfreeProducts	satisAnimalProducts	\
count	301.000000	209.000000	307.000000	
mean	6.767442	6.315789	7.348534	
std	1.981347	2.269317	1.902618	
min	1.000000	1.000000	1.000000	
25%	6.000000	5.000000	6.500000	
50%	7.000000	6.000000	8.000000	
75%	8.000000	8.000000	9.000000	
max	10.000000	10.000000	10.000000	

	ideasExtendedBusiness	ideasHelpCarry	ideasCustomerCouncil	\
count	324.000000	322.000000	318.000000	
mean	6.919753	3.711180	3.232704	
std	3.129760	3.027465	2.668179	
min	1.000000	1.000000	1.000000	
25%	5.000000	1.000000	1.000000	
50%	8.000000	2.000000	2.000000	
75%	10.000000	6.000000	5.000000	
max	10.000000	10.000000	10.000000	

	ideasFreeWifi	ideasTouchDisplay	ideasSelfCheckout	ideasBikeParking	\
count	324.000000	320.000000	323.000000	312.000000	
mean	6.410494	5.571875	7.857585	7.602564	
std	3.147757	3.197936	2.668804	2.752793	
min	1.000000	1.000000	1.000000	1.000000	
25%	4.000000	3.000000	7.000000	6.000000	
50%	7.000000	6.000000	9.000000	8.000000	
75%	9.000000	9.000000	10.000000	10.000000	
max	10.000000	10.000000	10.000000	10.000000	

```
       ideasUndergroundParking
count               300.000000
mean                  5.396667
std                   3.321057
min                   1.000000
25%                   2.000000
50%                   6.000000
75%                   8.000000
max                  10.000000
```

[8]:
```
#we can see that the column "income" has a high standard diviation and even␣
↪negative values. In case that this column
#is needed for further analysis cleaning is required
```

5 Data cleaning

[9]:
```
#Since we have a lot of different data in this data set I want to focus on the␣
↪question whether the expansion into
#online shopping is recommended for our shop. Therefore I want to analyse the␣
↪variable "orderingItems" and which other
#variables influence this. The first guess is that the variables
#"income", "distance", "modeOfTransportation", "frequency", "moneySpent",␣
↪"age", "findProducts"
#might have a correlation with the preference of ordering online or not.
#However the column "income" is as discussed above very messy. Since the survey␣
↪was fictional and it was not said if the
#income per month or per year is meant, it is difficult to handle this column.␣
↪Therefore we cannot respect the income in
#our analysis
```

[10]:
```
#first we want to rename the column with the amount of people to make the␣
↪dataset more intuitive even if we don't do further analysis on this column
df.rename(columns = {'G03Q13amountOfPeople':'amountOfPeople'}, inplace = True)
```

[11]:
```
#Furthermore we have to deal with missing values in the columns we want to␣
↪analyse. Since there were always only a few missing
#values we decide to delete them
df = df.dropna(subset=["orderingItems", "distance", "modeOfTransportation",␣
↪"frequency", "moneySpent", "age", "findProducts"])
```

[12]:
```
#now we need to encode the categorical values to make it easier to analyse and␣
↪to build a ML model
#we save the encoded data in a copy of the data frame to not loose the meaning␣
↪of the categorical data
df_encoded = df.copy()
```

```
df_encoded["distance"] = df_encoded["distance"].replace({"Less than few hundred␣
↪meters":0, "500 meters to 1km":1, "1-2km":2, "3-5km":3, "5-7km":4, ">7km":5})
df_encoded["modeOfTransportation"] = df_encoded["modeOfTransportation"] .
↪replace({"Walking":0, "Bicycle":1, "Own Car":2, "Public transportation":3,␣
↪"Rented car ("car sharing")":4, "Taxi":5})
df_encoded["frequency"] = df_encoded["frequency"].replace({"Once":0, "Twice":1,␣
↪"Three times":2, "Four times":3, "More than four times":4})
df_encoded["moneySpent"] = df_encoded["moneySpent"].replace({"Less than 25 USD":
↪0, "Between 25 and 50 USD":1, "Between 50 and 75":2, "Between 75 and 100␣
↪USD":3, "100 to 125 USD":4, "More than 125 USD":5})
#df_encoded["amountOfPeople"] = df_encoded["amountOfPeople"].replace({"1":0,␣
↪"2":1, "3":2, "4":3, "5":4, "5 or more":5})
df_encoded["age"] = df_encoded["age"].replace({"15-20":0, "20-25":1, "25-30":2,␣
↪"30-35":3, "35-40":4, "40-45":5, "45-50":6, "50-55":7, "55-60":8, "60-65":9,␣
↪"65-70":9, "70-75":10, ">75":11})
df_encoded["orderingItems"] = df_encoded["orderingItems"].replace({"…selecting␣
↪them myself in the store.":0, "… ordering online.":1})
df_encoded["findProducts"] = df_encoded["findProducts"].replace({"Strongly␣
↪agree":0, "Rather agree":1, "Neutral / Undecided":2, "Rather disagree":3,␣
↪"Strongly disagree":4})
```

[13]:
```
#check if the changes were successful
df_encoded.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 309 entries, 4 to 351
Data columns (total 46 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   randomInt             309 non-null    int64
 1   age                   309 non-null    int64
 2   gender                309 non-null    object
 3   district              303 non-null    object
 4   modeOfTransportation  309 non-null    int64
 5   distance              309 non-null    int64
 6   amountOfPeople        309 non-null    object
 7   income                297 non-null    float64
 8   frequency             309 non-null    int64
 9   days[1]               309 non-null    object
 10  days[2]               309 non-null    object
 11  days[3]               309 non-null    object
 12  days[4]               309 non-null    object
 13  days[5]               309 non-null    object
 14  days[6]               309 non-null    object
 15  days[7]               309 non-null    object
 16  time[1]               309 non-null    object
 17  time[2]               309 non-null    object
```

```
 18  time[3]                  309 non-null    object
 19  time[4]                  309 non-null    object
 20  time[5]                  309 non-null    object
 21  moneySpent               309 non-null    int64
 22  orderingItems            309 non-null    int64
 23  deliveringItems          304 non-null    object
 24  willingPayDelivery       153 non-null    object
 25  findProducts             309 non-null    int64
 26  usingDiscounts           303 non-null    object
 27  preferCash               307 non-null    object
 28  preferCashless           304 non-null    object
 29  isRelaxing               302 non-null    object
 30  satisGeneralStore        308 non-null    float64
 31  satisMusic               264 non-null    float64
 32  satisQualityProducts     303 non-null    float64
 33  satisGeneralAssortment   304 non-null    float64
 34  satisVeganProducts       252 non-null    float64
 35  satisOrganicProducts     278 non-null    float64
 36  satisGlutenfreeProducts  190 non-null    float64
 37  satisAnimalProducts      284 non-null    float64
 38  ideasExtendedBusiness    303 non-null    float64
 39  ideasHelpCarry           301 non-null    float64
 40  ideasCustomerCouncil     298 non-null    float64
 41  ideasFreeWifi            300 non-null    float64
 42  ideasTouchDisplay        299 non-null    float64
 43  ideasSelfCheckout        299 non-null    float64
 44  ideasBikeParking         291 non-null    float64
 45  ideasUndergroundParking  279 non-null    float64
dtypes: float64(17), int64(8), object(21)
memory usage: 113.5+ KB
```

[14]:
```
df_encoded.head()
```

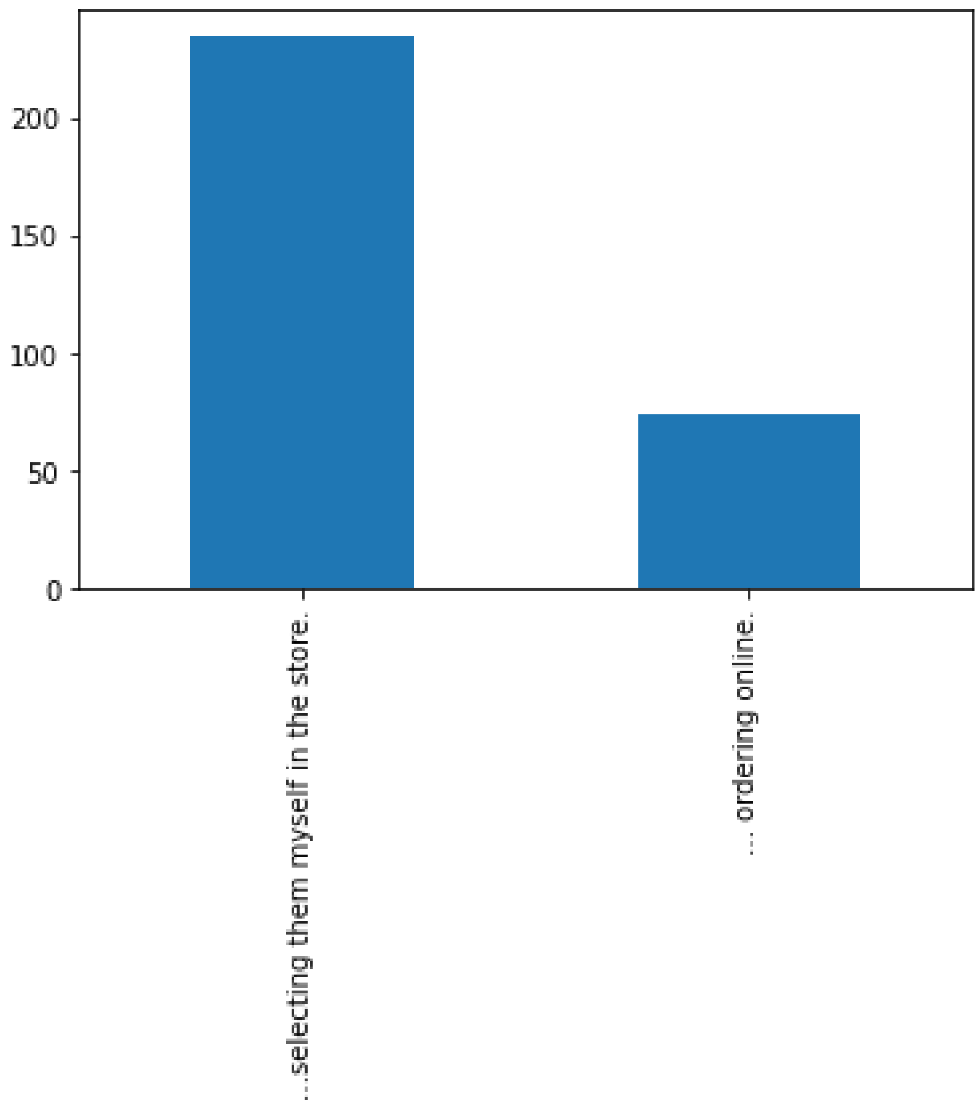
[14]:
```
   randomInt  age             gender    district  modeOfTransportation  \
4          3    0               Male   Piltunder                     2
5          3    1  Prefer not to say  Metrapalis                     0
6          2    9               Male      Godham                     2
7          1    5             Female      Godham                     1
8          1    2               Male  Metrapalis                     0

   distance amountOfPeople    income  frequency days[1]  …  \
4         2              4  250000.0          1      No  …
5         1              1     500.0          1      No  …
6         2              2    5000.0          0      No  …
7         1              1       NaN          1      No  …
8         1              1     600.0          3     Yes  …
```

```
  satisGlutenfreeProducts satisAnimalProducts ideasExtendedBusiness  \
4                      8.0                 1.0                   9.0
5                     10.0                10.0                   9.0
6                      NaN                 7.0                   5.0
7                      NaN                 7.0                   8.0
8                      6.0                 8.0                  10.0

  ideasHelpCarry ideasCustomerCouncil ideasFreeWifi ideasTouchDisplay  \
4            2.0                  1.0          10.0              10.0
5            1.0                  1.0           9.0               9.0
6            2.0                  2.0           6.0               3.0
7            3.0                  6.0          10.0               8.0
8            2.0                  3.0           4.0               3.0

  ideasSelfCheckout ideasBikeParking ideasUndergroundParking
4              10.0              8.0                     NaN
5              10.0              1.0                     1.0
6               9.0              9.0                     9.0
7              10.0              9.0                     1.0
8              10.0             10.0                     2.0

[5 rows x 46 columns]
```

6 EDA

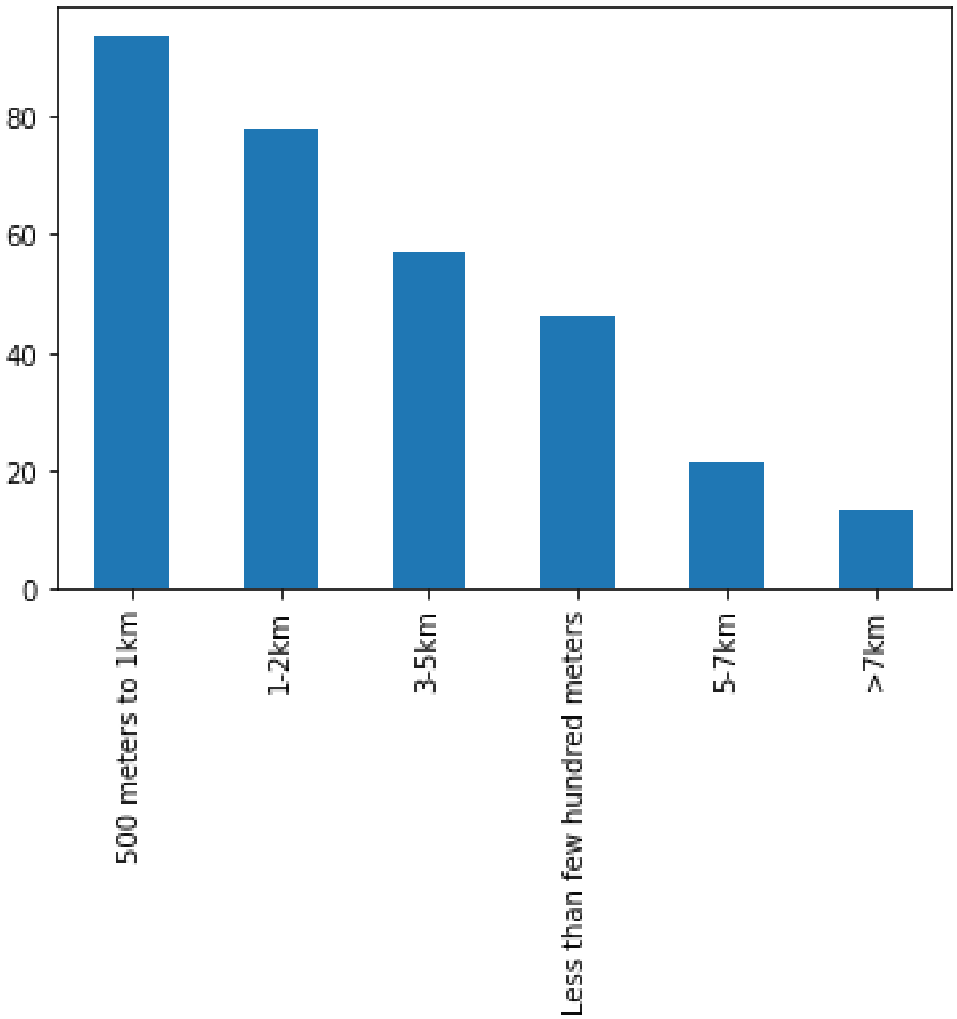
[15]:
```python
#we start with analysing the attitude of the customers towards online shopping
df.orderingItems.value_counts().plot(kind='bar')
countValues0 = df_encoded.orderingItems.value_counts()[0]
countValues1 = df_encoded.orderingItems.value_counts()[1]
print(str(round(countValues0/(countValues0+countValues1)*100)) +"% prefer␣
↪ordering in the store, " + str(round(countValues1/
↪(countValues0+countValues1)*100)) + "% prefer ordering online")
```

```
76% prefer ordering in the store, 24% prefer ordering online
```



[16]:
```
#Since most of the existing customers prefere going to the shop themeselves
#expanding the online shop might only be recommended to gain new customers.
#Therefore we continue with plotting the number of customers for each distance.
df.distance.value_counts().plot(kind='bar')
```

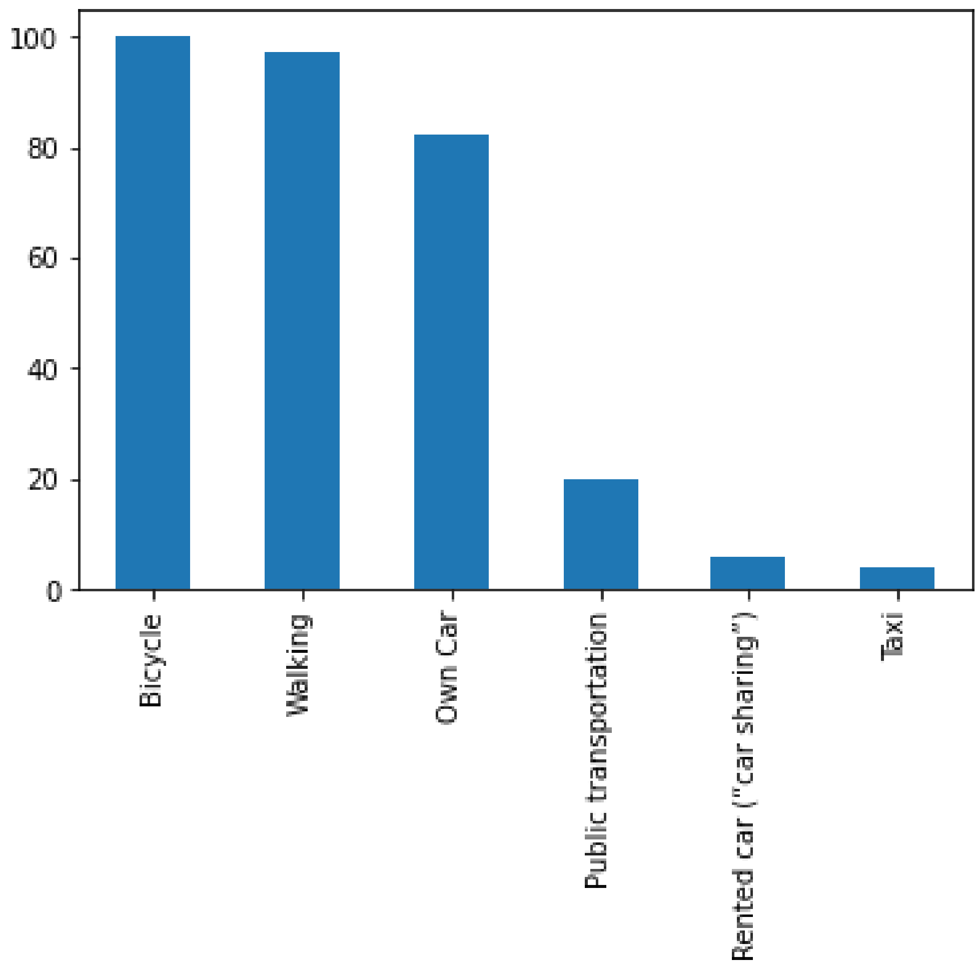
[16]: <AxesSubplot:>



[17]:
```
#Most people have a way of less than 2km to the shop. Therefore online shopping␣
↪might be a possibility to
#attract people who live further away
```

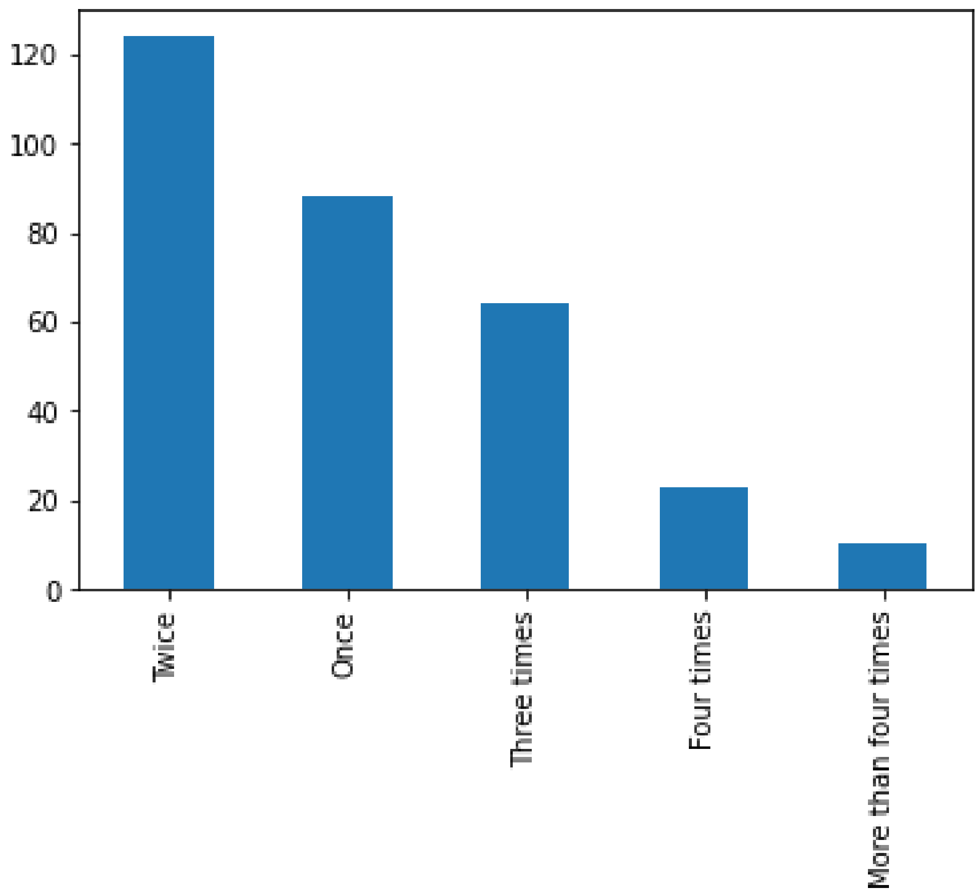
[18]:
```
#Another aspect that might make it more likely to do online shopping is the␣
↪mode of transportation you use for shopping
#We plot the amount of people for each mode of transportation
df.modeOfTransportation.value_counts().plot(kind='bar')
```

[18]: <AxesSubplot:>



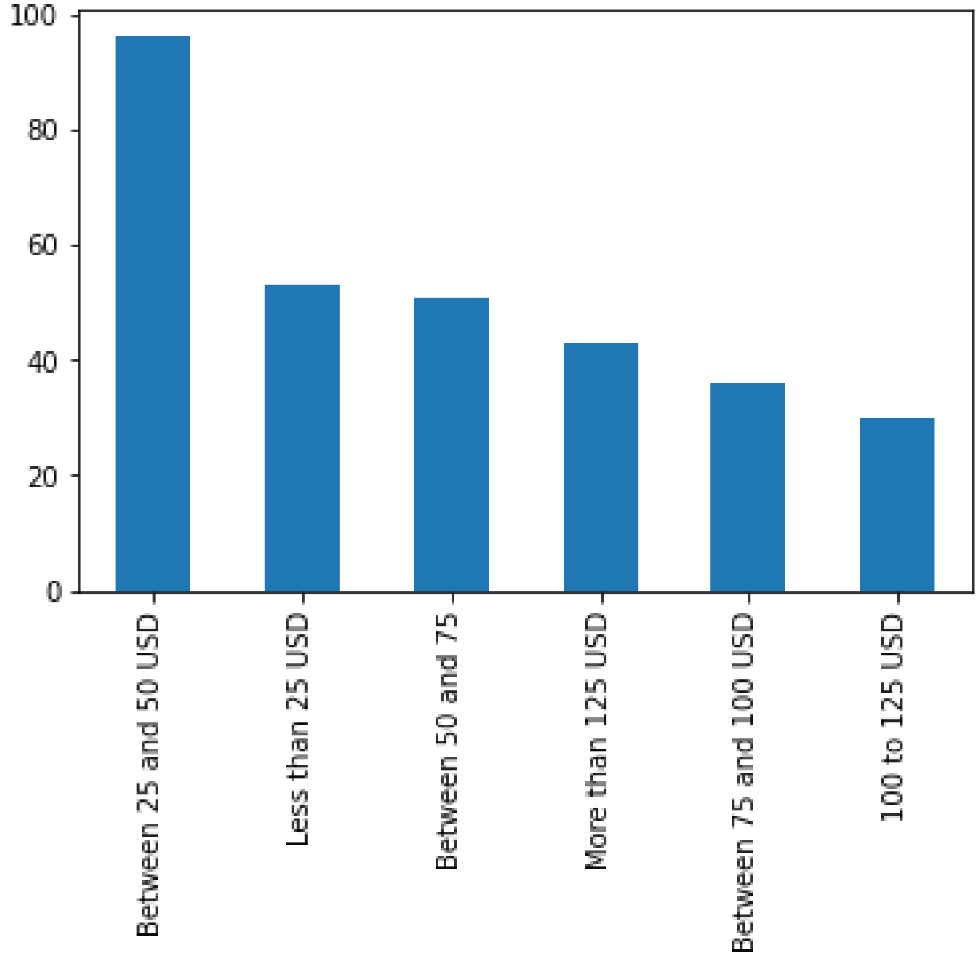
[19]:
```python
#We can see that the bike, walking and the own car are the most popular modes␣
↪of transportation, while the public transport
#or carsharing are less popular. People without an own car and without to go to␣
↪the shop by bike or walking might be
#attracted to online shopping
df.frequency.value_counts().plot(kind='bar')
```

[19]: <AxesSubplot:>



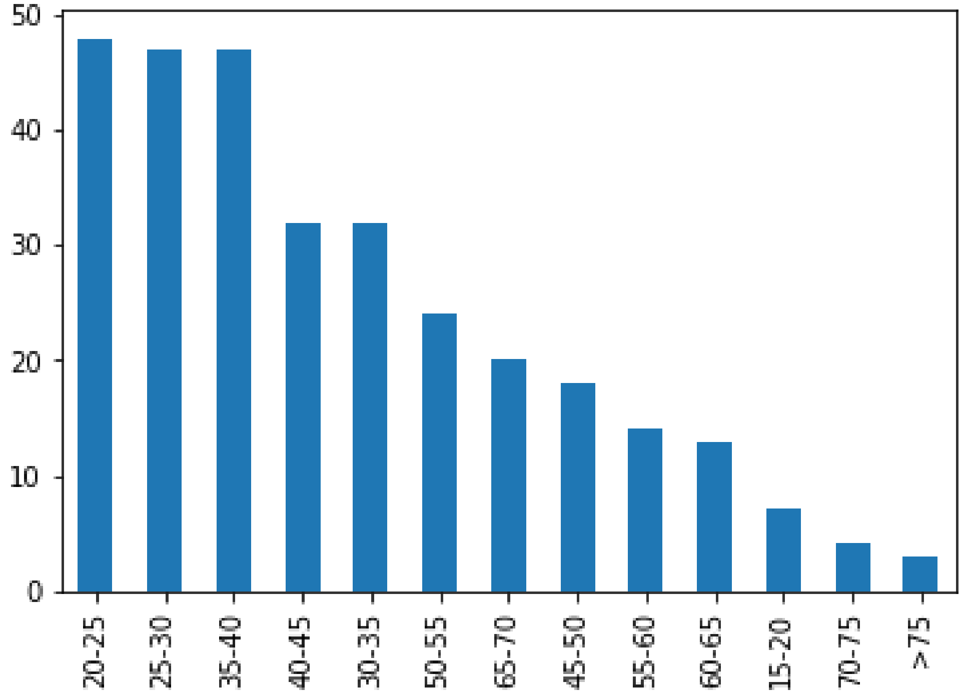

```
[20]: #Most people go less than 3 times a week to the supermarket
      #We want to find out how much the people spent. We expect that people who buy␣
       ↪more might tend to online shopping, since
      #they don't have to carry lots of groceries
      df.moneySpent.value_counts().plot(kind='bar')
```

```
[20]: <AxesSubplot:>
```



[21]:
```
#Most people pay a rather small amount of less than 50$.
#Furthermore we want to know how old our customers are since younger people␣
 ↪might be more attracted to modern shopping methods
df.age.value_counts().plot(kind='bar')
```

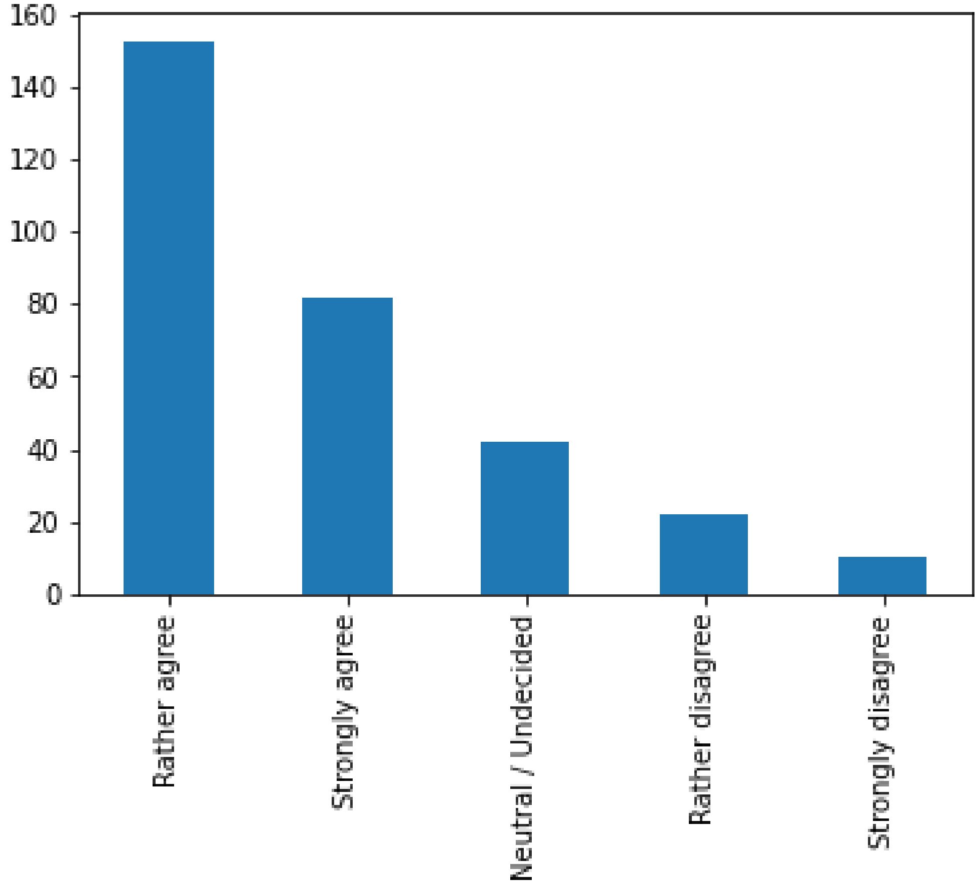
[21]: <AxesSubplot:>



[22]:
```
#compared to the general population our customers are very young which seems to␣
↪contradict the thesis that young people are
#more attracted to online shopping since only 24% of our young customership␣
↪said that they want to shop online
```

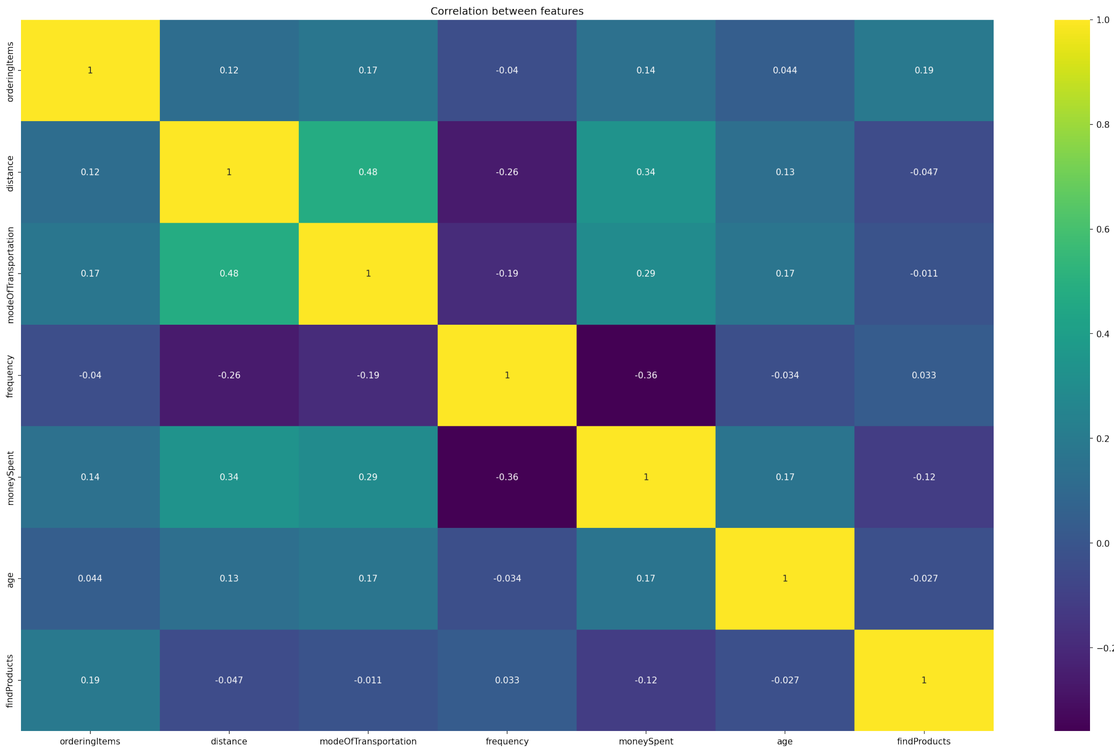
[23]:
```
#Last we want to find out how easy it is for our customers to find products in␣
↪the shop
df.findProducts.value_counts().plot(kind='bar')
```

[23]: <AxesSubplot:>



[24]:
```
#most people seem to be happy or very happy with finding products which might␣
↪explain why online shoipping is not
#really attractive to them
```

[25]:
```
#We now want to know in more detail how these variables influence each other␣
↪and print the correlation heatmap
plt.figure(figsize=(25,15),dpi=150)
sns.heatmap(df_encoded[["orderingItems", "distance", "modeOfTransportation",␣
↪"frequency", "moneySpent", "age", "findProducts"]].
↪corr(),cmap='viridis',annot=True)
plt.title('Correlation between features');
```




```
[26]: #As we expected after our first part of EDA, most of the factors we have choosen only have a small correlation with the
#preference of online ore "normal" shopping. Especially "frequency" and "age" have almost no correation.

#Based on these findings we now want to build a model that classifies wheter a customer is more likely to buy online or do
#shopping in the shop. For our independent variables we choose "distance", "modeOfTransportation", "moneySpent", "findProducts"
```

7 Data Processing and normalization

```
[27]: X = df_encoded[["distance", "modeOfTransportation", "moneySpent", "findProducts"]] #define independent variables

y = df_encoded["orderingItems"] #define dependent variables
```

```
[28]: #we split the data in 25% testing and 75% training data, we use stratify to achive comparable sets)
X_train, X_test, y_train, y_test = train_test_split(X,
                                                    y,
                                                    test_size=0.25,
```

```
                                                    random_state=20)
```

[29]:
```
#removes the mean and scales each feature to unit variance
scale = StandardScaler()

X_train_scaled = scale.fit_transform(X_train)
X_test_scaled = scale.transform(X_test)
```

8 Creating ML model 1

[30]:
```
#Since we are dealing with a classification problem, a logistic Regression␣
 ↪Model is a first good choice for a model
```

[31]:
```
model = LogisticRegression() #initializing the model
```

[32]:
```
model.fit(X_train_scaled, y_train) #fitting the model with the scaled data
```

[32]:
```
LogisticRegression()
```

8.1 Prediction on Test data

[33]:
```
#we predict the values for the variable "moneySpent" on the scaled testing data
```

[34]:
```
y_pred1 = model.predict(X_test_scaled)
```

8.2 Model 1 Performance

[35]:
```
#we evaluate our model by calculating the accuracy score of the model
```

[36]:
```
print(str(metrics.accuracy_score(y_test, y_pred1) * 100)+"% accuracy")
```

```
67.94871794871796% accuracy
```

9 Creating ML model 2

[37]:
```
#Now we want to use a KNN model to compare it with the first one
```

[73]:
```
# we set neighbors that vote on a "new" entity to 6 -> showed best results in␣
 ↪testing
classifier = KNeighborsClassifier(n_neighbors=6)
```

[74]:
```
# Fitting the model
classifier.fit(X_train_scaled, y_train)
```

[74]:
```
KNeighborsClassifier(n_neighbors=6)
```

9.1 Prediction on Test data

[75]:
```
# Predicting the Test set results
y_pred2 = classifier.predict(X_test_scaled)
```

9.2 Model 2 Performance

[76]:
```
#we evaluate our model by calculating the accuracy score of the model
```

[77]:
```
print(str(metrics.accuracy_score(y_test, y_pred2) * 100)+"% accuracy")
```

```
71.7948717948718% accuracy
```

10 Report and insight from your analysis

[ ]:
```
#In this notebook we analysed the supermarket survey that was made during the␣
↪course. We focused on the question if an
#expansion of the online market is recommended or not. Therefore we analysed␣
↪the columns
#"orderingItems", "distance", "modeOfTransportation", "frequency",␣
↪"moneySpent", "age" and "findProducts".
```

[ ]:
```
#We came to the conclusion that around 76% of our customers prefer going to the␣
↪shop less than 3 times a week themselves.
#Each time most of them spent less than 50$ and live closer than 2km to the␣
↪market. Car, bike and walking are the most
#used mods of transportation. The great majority of the customers is happy with␣
↪finding products.
#Compared to the general population the customership is very young.
```

[ ]:
```
#However most of the analysed variables have a only a small correlation with␣
↪the prefered way of shopping.
#Age and frequency have almost no influence. A long distance to the shop,␣
↪having no one car or the possibility walk or
#use a bike, spending a higher sum of money and finding it less easy to find␣
↪products increase the likeiness of prefering
#online shopping.
```

[ ]:
```
#Build on these findings we wanted to train a model that classifies whether a␣
↪customer rather prefers going to the shop or
#shopping online. We chose a Logistic Regression Model and a KNN model which␣
↪delivered around 68% respectively 72% accuracy.
#These results are not really satisfying since 76% prefer going to the shop␣
↪themselves. Always predicting this result delivers
#a higher accuracy.
```

[ ]:
```
#Lastly we have to say that the data is not really representative. Although␣
↪every participant had a scenario in which
#they should put themeselves it is very propable that the dataset is biased␣
↪towards the peergroupe of this course. This makes
#it difficult to analyse the dataset because it makes it hard to find␣
↪corrolations since all data points are quite similar
#to each other.
```