In [406]:

```
#Vraag 1
import scipy.stats as stats
from scipy.stats import norm
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
from sklearn import tree
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.naive_bayes import BernoulliNB
from sklearn.svm import LinearSVC
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

#avg = 64
#std = 8

#P(Z > a) = 0.7

print("Hij moet minstens ", stats.norm.ppf(1 - 0.7, 64, 8), " halen")
```

```
Hij moet minstens  59.804795898335676  halen
```

In [407]:

```
#Vraag 1 b

print(((1 - stats.norm.cdf(72, 64, 8)) * 100), "% van de kanditaten behalen een hogere scor
```

```
15.865525393145708 % van de kanditaten behalen een hogere score
```

In [408]:

```
games = pd.read_csv('Video_Games_Sales.csv')
```

In [409]:

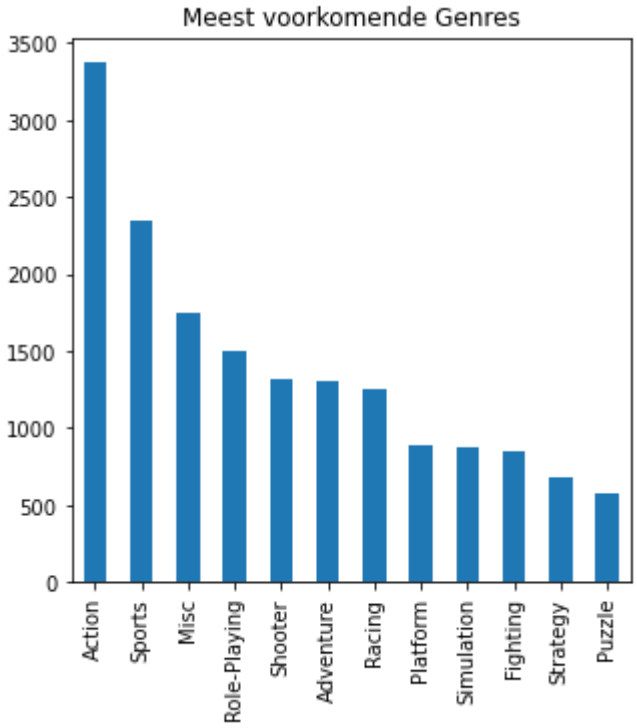
```
#Vraag 2
platform = games.Platform
games.Platform.value_counts()

#Antwoord: Er zijn 31 Platformen
```

Out[409]:

```
PS2     2161
DS      2152
PS3     1331
Wii     1320
X360    1262
PSP     1209
PS      1197
PC       974
XB       824
GBA      822
GC       556
3DS      520
PSV      432
PS4      393
N64      319
XOne     247
SNES     239
SAT      173
WiiU     147
2600     133
NES       98
GB        98
DC        52
GEN       29
NG        12
SCD        6
WS         6
3DO        3
TG16       2
GG         1
PCFX       1
Name: Platform, dtype: int64
```

In [410]:

```
#Vraag 3
bar_data = games.Genre.value_counts()
bar_plot = bar_data.plot.bar(title = 'Meest voorkomende Genres', figsize=(5,5))
```



In [411]:

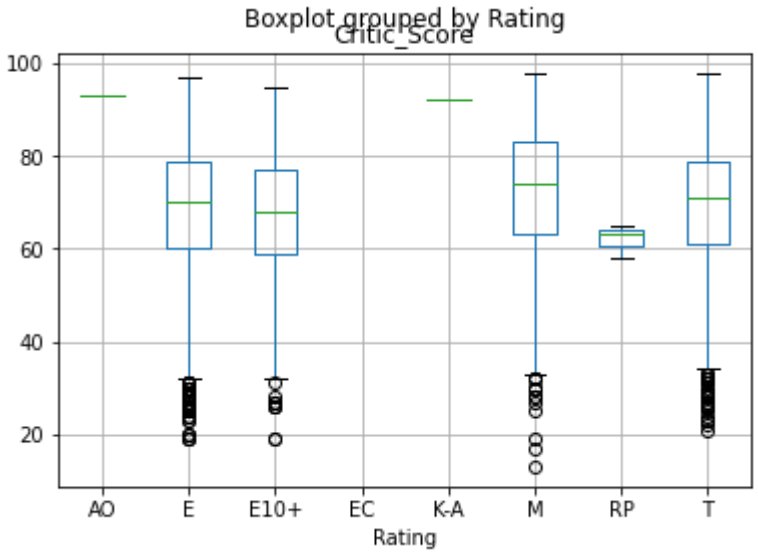
```
#Antwoord: De 5 meest voorkomende Genres zijn Action, Sports, Misc, Role-Playing, en Shoote
```

In [412]:

```
#Vraag 4
games[games.Rating.notnull()].boxplot('Critic_Score', 'Rating')
```

Out[412]:

```
<AxesSubplot:title={'center':'Critic_Score'}, xlabel='Rating'>
```



In [413]:

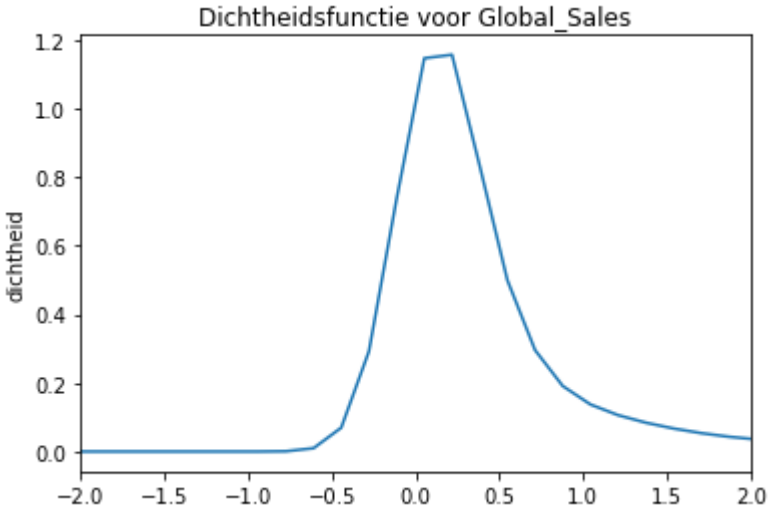
```
#Bespreking
#Er zou wel een verband kunnen zijn. Aan de hand van de bovenste data kunnen we zien dat Ra
#een hoge critic score krijgen meestal, terwijl Ratings zoals EC, K-A, en RP vaak minder in
#een lagere of zelfs geen Critic_Score krijgen.
```

In [414]:

```
#Vraag 5

games.Global_Sales.plot(kind = 'kde', title = 'Dichtheidsfunctie voor Global_Sales')
plt.ylabel("dichtheid")
ax = plt.gca() #Get the Current Axes
ax.set_xlim(-2, 2)
```

Out[414]:

```
(-2.0, 2.0)
```

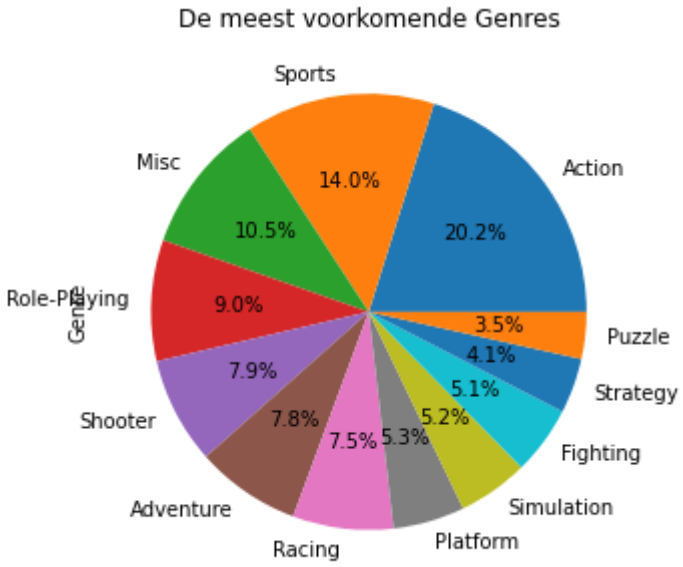


In [415]:

```
#Antwoord: De dichtheid van Global_Sales is een beetje rechtsscheef
```

In [416]:

```
#Vraag 6
pie_data = games['Genre'].value_counts()
pie_plot = pie_data.plot.pie(title = 'De meest voorkomende Genres', figsize=(5,5), autopct

#Antwoord: Aan de hand van de Pie Chart kunnen we aflezen dat Action de Genre is die het me
```



In [417]:

```
#Vraag 7
games.Rating.value_counts()
```

Out[417]:

```
E       3991
T       2961
M       1563
E10+    1420
EC         8
K-A        3
RP         3
AO         1
Name: Rating, dtype: int64
```

In [418]:

```
# De meest voorkomende Rating is: 'E', dus die gaan wij gebruiken:

games.loc[games.Rating.isnull(), 'Rating'] = 'E'
```

In [419]:

```
#Vraag 8

games_ML = pd.DataFrame(data = games, columns = ['Year_of_Release', 'Genre', 'Critic_Score'
games_ML['target'] = games.Global_Sales
```

In [420]:

```
#Vraag 9
games_ML_zonder_missing = games_ML.dropna()
print(games_ML_zonder_missing.shape)
print(games_ML_zonder_missing.notnull().sum())

#Omdat de shape van games_ML_zonder_missing.shape (7983,4) is, en dat overeenkomt met de so
#zijn alle waarden notnull.
```

```
(7983, 4)
Year_of_Release    7983
Genre              7983
Critic_Score       7983
target             7983
dtype: int64
```

In [421]:

```
#Vraag 10 ik zal eerst nagaan welke feature niet gebruik wordt. Ik neem aan dat het Genre i
#en de ML_algoritmes moeite zullen hebben hiermee.
```

In [422]:

```
games_ML_zonder_missing.Genre.value_counts()
```

Out[422]:

```
Action          1851
Sports          1165
Shooter          923
Role-Playing     731
Racing           725
Misc             509
Platform         490
Fighting         405
Simulation       348
Adventure        320
Strategy         296
Puzzle           220
Name: Genre, dtype: int64
```

In [423]:

```
Action = games_ML_zonder_missing[games_ML_zonder_missing.Genre == 'Action']
Sports = games_ML_zonder_missing[games_ML_zonder_missing.Genre == 'Sports']
Shooter = games_ML_zonder_missing[games_ML_zonder_missing.Genre == 'Shooter']
role_playing = games_ML_zonder_missing[games_ML_zonder_missing.Genre == 'Role-Playing']
Racing = games_ML_zonder_missing[games_ML_zonder_missing.Genre == 'Racing']
Misc = games_ML_zonder_missing[games_ML_zonder_missing.Genre == 'Misc']
Platform = games_ML_zonder_missing[games_ML_zonder_missing.Genre == 'Platform']
Fighting = games_ML_zonder_missing[games_ML_zonder_missing.Genre == 'Fighting']
Simulation = games_ML_zonder_missing[games_ML_zonder_missing.Genre == 'Simulation']
Adventure = games_ML_zonder_missing[games_ML_zonder_missing.Genre == 'Adventure']
Strategy = games_ML_zonder_missing[games_ML_zonder_missing.Genre == 'Strategy']
Puzzle = games_ML_zonder_missing[games_ML_zonder_missing.Genre == 'Puzzle']

games_ML_zonder_missing.Action[games_ML_zonder_missing.Genre] = 1
games_ML_zonder_missing.Sports[games_ML_zonder_missing.Genre] = 2
games_ML_zonder_missing.Shooter[games_ML_zonder_missing.Genre] = 3
games_ML_zonder_missing.role_playing[games_ML_zonder_missing.Genre] = 4
games_ML_zonder_missing.Racing[games_ML_zonder_missing.Genre] = 5
games_ML_zonder_missing.Misc[games_ML_zonder_missing.Genre] = 6
games_ML_zonder_missing.Platform[games_ML_zonder_missing.Genre] = 7
games_ML_zonder_missing.Fighting[games_ML_zonder_missing.Genre] = 8
games_ML_zonder_missing.Simulation[games_ML_zonder_missing.Genre] = 9
games_ML_zonder_missing.Adventure[games_ML_zonder_missing.Genre] = 10
games_ML_zonder_missing.Strategy[games_ML_zonder_missing.Genre] = 11
games_ML_zonder_missing.Puzzle[games_ML_zonder_missing.Genre] = 12


#niet helemaal gelukt helaas
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_1400/1454009207.py in <module>
     12 Puzzle = games_ML_zonder_missing[games_ML_zonder_missing.Genre == 'P
uzzle']
     13
---> 14 games_ML_zonder_missing.Action[games_ML_zonder_missing.Genre] = 1
     15 games_ML_zonder_missing.Sports[games_ML_zonder_missing.Genre] = 2
     16 games_ML_zonder_missing.Shooter[games_ML_zonder_missing.Genre] = 3

~\anaconda3\lib\site-packages\pandas\core\generic.py in __getattr__(self, na
me)
   5485         ):
   5486             return self[name]
-> 5487         return object.__getattribute__(self, name)
   5488
   5489     def __setattr__(self, name: str, value) -> None:

AttributeError: 'DataFrame' object has no attribute 'Action'
```

In [424]:

```
games_ML_zonder_missing.head()
```

Out[424]:

	Year_of_Release	Genre	Critic_Score	target
0	2006.0	Sports	76.0	82.53
2	2008.0	Racing	82.0	35.52
3	2009.0	Sports	80.0	32.77
6	2006.0	Platform	89.0	29.80
7	2006.0	Misc	58.0	28.92

In [425]:

```
#Vraag 11

X=games_ML_zonder_missing
y=games_ML_zonder_missing.target
```

In [426]:

```
#Vraag 12
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25)
print(X_train.shape)
print(y_train.shape)
print(X_test.shape)
print(y_test.shape)
```

```
(5987, 4)
(5987,)
(1996, 4)
(1996,)
```

In [427]:

```
#Vraag 13

model = tree.DecisionTreeClassifier(random_state=0, max_depth = 3)   #set tree levels with
model = model.fit(X, y)
model = model.fit(X_train, y_train)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_1400/3300852135.py in <module>
      2
      3 model = tree.DecisionTreeClassifier(random_state=0, max_depth = 3)
#set tree levels with max_depth
----> 4 model = model.fit(X, y)
      5 model = model.fit(X_train, y_train)

~\anaconda3\lib\site-packages\sklearn\tree\_classes.py in fit(self, X, y, sa
mple_weight, check_input, X_idx_sorted)
    901         """
    902
--> 903         super().fit(
    904             X, y,
    905             sample_weight=sample_weight,

~\anaconda3\lib\site-packages\sklearn\tree\_classes.py in fit(self, X, y, sa
mple_weight, check_input, X_idx_sorted)
    155             check_X_params = dict(dtype=DTYPE, accept_sparse="csc")
    156             check_y_params = dict(ensure_2d=False, dtype=None)
--> 157             X, y = self._validate_data(X, y,
    158                                        validate_separately=(check_X_
params,
    159                                                             check_y_
params))

~\anaconda3\lib\site-packages\sklearn\base.py in _validate_data(self, X, y,
 reset, validate_separately, **check_params)
    428                 # :(
    429                 check_X_params, check_y_params = validate_separately
--> 430                 X = check_array(X, **check_X_params)
    431                 y = check_array(y, **check_y_params)
    432             else:

~\anaconda3\lib\site-packages\sklearn\utils\validation.py in inner_f(*args,
 **kwargs)
     61             extra_args = len(args) - len(all_args)
     62             if extra_args <= 0:
---> 63                 return f(*args, **kwargs)
     64
     65             # extra_args > 0

~\anaconda3\lib\site-packages\sklearn\utils\validation.py in check_array(arr
ay, accept_sparse, accept_large_sparse, dtype, order, copy, force_all_finit
e, ensure_2d, allow_nd, ensure_min_samples, ensure_min_features, estimator)
    671                     array = array.astype(dtype, casting="unsafe", co
py=False)
    672                 else:
--> 673                     array = np.asarray(array, order=order, dtype=dty
pe)
    674             except ComplexWarning as complex_warning:
    675                 raise ValueError("Complex data not supported\n"
```

```
~\anaconda3\lib\site-packages\numpy\core\_asarray.py in asarray(a, dtype, or
der, like)
    100         return _asarray_with_like(a, dtype=dtype, order=order, like=
like)
    101
--> 102     return array(a, dtype, copy=False, order=order)
    103
    104

~\anaconda3\lib\site-packages\pandas\core\generic.py in __array__(self, dtyp
e)
   1991
   1992     def __array__(self, dtype: NpDtype | None = None) -> np.ndarray:
-> 1993         return np.asarray(self._values, dtype=dtype)
   1994
   1995     def __array_wrap__(

~\anaconda3\lib\site-packages\numpy\core\_asarray.py in asarray(a, dtype, or
der, like)
    100         return _asarray_with_like(a, dtype=dtype, order=order, like=
like)
    101
--> 102     return array(a, dtype, copy=False, order=order)
    103
    104

ValueError: could not convert string to float: 'Sports'
```

In [428]:

```
#Vraag 14:
y_pred = model.predict(X_test)
print("Accuracy = ", accuracy_score(y_test, y_pred) * 100, "%")
```

```
---------------------------------------------------------------------------
NotFittedError                            Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_1400/318514267.py in <module>
      1 #Vraag 14:
----> 2 y_pred = model.predict(X_test)
      3 print("Accuracy = ", accuracy_score(y_test, y_pred) * 100, "%")

~\anaconda3\lib\site-packages\sklearn\tree\_classes.py in predict(self, X, c
heck_input)
    439             The predicted classes, or the predict values.
    440         """
--> 441         check_is_fitted(self)
    442         X = self._validate_X_predict(X, check_input)
    443         proba = self.tree_.predict(X)

~\anaconda3\lib\site-packages\sklearn\utils\validation.py in inner_f(*args,
 **kwargs)
     61             extra_args = len(args) - len(all_args)
     62             if extra_args <= 0:
---> 63                 return f(*args, **kwargs)
     64
     65             # extra_args > 0

~\anaconda3\lib\site-packages\sklearn\utils\validation.py in check_is_fitted
(estimator, attributes, msg, all_or_any)
   1096
   1097     if not attrs:
-> 1098         raise NotFittedError(msg % {'name': type(estimator).__name__
})
   1099
   1100

NotFittedError: This DecisionTreeClassifier instance is not fitted yet. Call
'fit' with appropriate arguments before using this estimator.
```

In [ ]:

```

```