Московский государственный технический университет им. Н.Э. Баумана
Факультет «Информатика и системы управления»
Кафедра «Автоматизированные системы обработки информации и управления»

Отчет по лабораторной работе № 3

«Обработка пропусков в данных, кодирование категориальных признаков, масштабирование данных.»

По курсу «Технологии машинного обучения»

ИСПОЛНИТЕЛЬ:

Кционова А. А.
Группа ИУ5-63

_____

"__"_____2019 г.

ПРЕПОДАВАТЕЛЬ:

Гапанюк Ю. Е.

_____

"__"_____2019 г.

Москва 2019

Цель лабораторной работы.

Изучить способы предварительной обработки данных для дальнейшего формирования моделей.

Практическая часть.

```
import pandas as pd
import numpy as np
from sklearn.impute import SimpleImputer, MissingIndicator
from sklearn.preprocessing import LabelEncoder, OneHotEncoder, MinMaxScaler,
StandardScaler, Normalizer

import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
```

Характеристика выбранного датасета:

page_id | The unique identifier for that characters page within the wikia
name | The name of the character
urlslug | The unique url within the wikia that takes you to the character
ID | The identity status of the character (Secret Identity, Public identity, [on marvel only: No Dual Identity])
ALIGN | If the character is Good, Bad or Neutral
EYE | Eye color of the character
HAIR | Hair color of the character
SEX | Sex of the character (e.g. Male, Female, etc.)
GSM | If the character is a gender or sexual minority (e.g. Homosexual characters, bisexual characters)
ALIVE | If the character is alive or deceased
APPEARANCES | The number of appareances of the character in comic books (as of Sep. 2, 2014. Number will become increasingly out of date as time goes on.)
FIRST APPEARANCE | The month and year of the character's first appearance in a comic book, if available
YEAR | The year of the character's first appearance in a comic book, if available

In [251]:

```
dc = pd.read_csv('C:/Users/kotsi/dc-wikia-data.csv')
```

1.Обработка пропусков данных

In [252]:

```
dc.head()
```

Out[252]:

	page_id	name	urlslug	ID	ALIGN	EYE	HAIR	SEX	GSM	ALIVE	APPEARANCES	FIRST APPEARANCE	YEAR
0	1422	Batman (Bruce Wayne)	\/wiki\/Batman_(Bruce_Wayne)	Secret Identity	Good Characters	Blue Eyes	Black Hair	Male Characters	NaN	Living Characters	3093.0	1939, May	1939.0

	page_id	name	urlslug	ID	ALIGN	EYE	HAIR	SEX	GSM	ALIVE	APPEARANCES	FIRST APPEARANCE	YEAR
1	23387	Superman (Clark Kent)	\/wiki\/Superman_(Clark_Kent)	Secret Identity	Good Characters	Blue Eyes	Black Hair	Male Characters	NaN	Living Characters	2496.0	1986, October	1986.0
2	1458	Green Lantern (Hal Jordan)	\/wiki\/Green_Lantern_(Hal_Jordan)	Secret Identity	Good Characters	Brown Eyes	Brown Hair	Male Characters	NaN	Living Characters	1565.0	1959, October	1959.0
3	1659	James Gordon (New Earth)	\/wiki\/James_Gordon_(New_Earth)	Public Identity	Good Characters	Brown Eyes	White Hair	Male Characters	NaN	Living Characters	1316.0	1987, February	1987.0
4	1576	Richard Grayson (New Earth)	\/wiki\/Richard_Grayson_(New_Earth)	Secret Identity	Good Characters	Blue Eyes	Black Hair	Male Characters	NaN	Living Characters	1237.0	1940, April	1940.0

In [253]:
```
dc.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6896 entries, 0 to 6895
Data columns (total 13 columns):
page_id              6896 non-null int64
name                 6896 non-null object
urlslug              6896 non-null object
ID                   4883 non-null object
ALIGN                6295 non-null object
EYE                  3268 non-null object
HAIR                 4622 non-null object
SEX                  6771 non-null object
GSM                  64 non-null object
ALIVE                6893 non-null object
APPEARANCES          6541 non-null float64
FIRST APPEARANCE     6827 non-null object
YEAR                 6827 non-null float64
dtypes: float64(2), int64(1), object(10)
memory usage: 700.5+ KB
```

In [254]:
```
dc['FIRST APPEARANCE']=dc['FIRST APPEARANCE'].str[5:]
cat_cols = [c for c in dc.columns if dc[c].dtype.name == 'object']
num_cols=[c for c in dc.columns if dc[c].dtype.name != 'object']
```

In [255]:
```
dc[cat_cols].describe()
```
Out[255]:

	name	urlslug	ID	ALIGN	EYE	HAIR	SEX	GSM	ALIVE	FIRST APPEARANCE
count	6896	6896	4883	6295	3268	4622	6771	64	6893	6827
unique	6896	6896	3	4	17	17	4	2	2	14
top	Ernest Widdle (New Earth)	\/wiki\/Rees-Van_(New_Earth)	Public Identity	Bad Characters	Blue Eyes	Black Hair	Male Characters	Homosexual Characters	Living Characters	August
freq	1	1	2466	2895	1102	1574	4783	54	5200	634

In [256]:
```
cat_dc=dc[cat_cols]
cat_dc.head()
```
Out[256]:

	name	urlslug	ID	ALIGN	EYE	HAIR	SEX	GSM	ALIVE	FIRST APPEARANCE
0	Batman (Bruce Wayne)	\/wiki\/Batman_(Bruce_Wayne)	Secret Identity	Good Characters	Blue Eyes	Black Hair	Male Characters	NaN	Living Characters	May
1	Superman (Clark Kent)	\/wiki\/Superman_(Clark_Kent)	Secret Identity	Good Characters	Blue Eyes	Black Hair	Male Characters	NaN	Living Characters	October
2	Green Lantern (Hal Jordan)	\/wiki\/Green_Lantern_(Hal_Jordan)	Secret Identity	Good Characters	Brown Eyes	Brown Hair	Male Characters	NaN	Living Characters	October
3	James Gordon (New Earth)	\/wiki\/James_Gordon_(New_Earth)	Public Identity	Good Characters	Brown Eyes	White Hair	Male Characters	NaN	Living Characters	February
4	Richard Grayson (New Earth)	\/wiki\/Richard_Grayson_(New_Earth)	Secret Identity	Good Characters	Blue Eyes	Black Hair	Male Characters	NaN	Living Characters	April

In [258]:
```
num_dc=dc[num_cols]
```

```
num_dc.head()
```

Out[258]:

	page_id	APPEARANCES	YEAR
0	1422	3093.0	1939.0
1	23387	2496.0	1986.0
2	1458	1565.0	1959.0
3	1659	1316.0	1987.0
4	1576	1237.0	1940.0

In [259]:

```
dc.groupby('ID')['ID'].count()
```

Out[259]:

```
ID
Identity Unknown        9
Public Identity      2466
Secret Identity      2408
Name: ID, dtype: int64
```

In [260]:

```
dc.loc[dc["ID"].isnull()].head()
```

Out[260]:

	page_id	name	urlslug	ID	ALIGN	EYE	HAIR	SEX	GSM	ALIVE	APPEARANCES	FIRST APPEARANCE	YEAR
181	17885	Samuel Morgan (New Earth)	\/wiki\/Samuel_Morgan_(New_Earth)	NaN	Good Characters	NaN	Red Hair	Male Characters	NaN	Deceased Characters	155.0	March	1937.0
203	113540	Tubby Watts (New Earth)	\/wiki\/Tubby_Watts_(New_Earth)	NaN	Good Characters	NaN	Red Hair	Male Characters	NaN	Living Characters	137.0	September	1941.0

	page_id	name	urlslug	ID	ALIGN	EYE	HAIR	SEX	GSM	ALIVE	APPEARANCES	FIRST APPEARANCE	YEAR
220	6063	Victory (New Earth)	\/wiki\/Victory_(New_Earth)	NaN	Good Characters	Black Eyes	White Hair	Male Characters	NaN	Living Characters	130.0	September	1941.0
251	1841	Dolphin (New Earth)	\/wiki\/Dolphin_(New_Earth)	NaN	Good Characters	Blue Eyes	Silver Hair	Female Characters	NaN	Deceased Characters	115.0	December	1968.0
277	341732	Sugar Plumm (New Earth)	\/wiki\/Sugar_Plumm_(New_Earth)	NaN	Good Characters	NaN	Blond Hair	Female Characters	NaN	Living Characters	105.0	May	1956.0

Личность всех этих персонажей можно охарактеризовать как неизвестную

In [261]:
```
dc['ID']=dc["ID"].fillna('Identity Unknown')
```

In [262]:
```
dc.groupby('ALIGN')['ALIGN'].count()
```

Out[262]:
```
ALIGN
Bad Characters        2895
Good Characters       2832
Neutral Characters     565
Reformed Criminals       3
Name: ALIGN, dtype: int64
```

In [263]:
```
dc['ALIGN']=dc["ALIGN"].fillna('Neutral Characters')
```

In [264]:
```
dc.groupby('ALIGN')['ALIGN'].count()
```

Out[264]:
```
ALIGN
Bad Characters        2895
Good Characters       2832
Neutral Characters    1166
Reformed Criminals       3
Name: ALIGN, dtype: int64
```

Следующие признаки либо маловажны, либо заполнены практически целиком, так что их мы заполним наиболее частыми

In [265]:

```
data_describe = dc.describe(include=[object])
for c in ['EYE','HAIR','SEX','ALIVE','FIRST APPEARANCE']:
    dc[c] = dc[c].fillna(data_describe[c]['top'])
dc.head()
```

Out[265]:

	page_id	name	urlslug	ID	ALIGN	EYE	HAIR	SEX	GSM	ALIVE	APPEARANCES	FIRST APPEARANCE	YEAR
0	1422	Batman (Bruce Wayne)	\/wiki\/Batman_(Bruce_Wayne)	Secret Identity	Good Characters	Blue Eyes	Black Hair	Male Characters	NaN	Living Characters	3093.0	May	1939.0
1	23387	Superman (Clark Kent)	\/wiki\/Superman_(Clark_Kent)	Secret Identity	Good Characters	Blue Eyes	Black Hair	Male Characters	NaN	Living Characters	2496.0	October	1986.0
2	1458	Green Lantern (Hal Jordan)	\/wiki\/Green_Lantern_(Hal_Jordan)	Secret Identity	Good Characters	Brown Eyes	Brown Hair	Male Characters	NaN	Living Characters	1565.0	October	1959.0
3	1659	James Gordon (New Earth)	\/wiki\/James_Gordon_(New_Earth)	Public Identity	Good Characters	Brown Eyes	White Hair	Male Characters	NaN	Living Characters	1316.0	February	1987.0
4	1576	Richard Grayson (New Earth)	\/wiki\/Richard_Grayson_(New_Earth)	Secret Identity	Good Characters	Blue Eyes	Black Hair	Male Characters	NaN	Living Characters	1237.0	April	1940.0

Введем гетеросексуальную ориентацию и предположим, что все остальные супергерои действительно таковы

In [266]:

```
dc['GSM']=dc["GSM"].fillna('Heterosexual Characters')
```

Посмотрим, все ли заполнили

In [267]:

```
dc.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6896 entries, 0 to 6895
Data columns (total 13 columns):
page_id             6896 non-null int64
name                6896 non-null object
urlslug             6896 non-null object
ID                  6896 non-null object
```

```
ALIGN                6896 non-null object
EYE                  6896 non-null object
HAIR                 6896 non-null object
SEX                  6896 non-null object
GSM                  6896 non-null object
ALIVE                6896 non-null object
APPEARANCES          6541 non-null float64
FIRST APPEARANCE     6896 non-null object
YEAR                 6827 non-null float64
dtypes: float64(2), int64(1), object(10)
memory usage: 700.5+ KB
```

Заполним числовые признаки

In [268]:

```
dc[num_cols] = dc[num_cols].fillna(dc[num_cols].median(axis=0), axis=0)
```

Проеведем еще небольшое преобразование данных, отсечем год в столбце "Первое появление" и преобразуем некоторые типы

In [270]:

```
dc[['YEAR','APPEARANCES']]=dc[['YEAR','APPEARANCES']].astype(int)
dc.head()
```

Out[270]:

	page_id	name	urlslug	ID	ALIGN	EYE	HAIR	SEX	GSM	ALIVE	APPEARANCES	FIRST APPEARANCE	YEAR
0	1422	Batman (Bruce Wayne)	\/wiki\/Batman_(Bruce_Wayne)	Secret Identity	Good Characters	Blue Eyes	Black Hair	Male Characters	Heterosexual Characters	Living Characters	3093	May	1939
1	23387	Superman (Clark Kent)	\/wiki\/Superman_(Clark_Kent)	Secret Identity	Good Characters	Blue Eyes	Black Hair	Male Characters	Heterosexual Characters	Living Characters	2496	October	1986
2	1458	Green Lantern (Hal Jordan)	\/wiki\/Green_Lantern_(Hal_Jordan)	Secret Identity	Good Characters	Brown Eyes	Brown Hair	Male Characters	Heterosexual Characters	Living Characters	1565	October	1959
3	1659	James Gordon (New Earth)	\/wiki\/James_Gordon_(New_Earth)	Public Identity	Good Characters	Brown Eyes	White Hair	Male Characters	Heterosexual Characters	Living Characters	1316	February	1987

	page_id	name	urlslug	ID	ALIGN	EYE	HAIR	SEX	GSM	ALIVE	APPEARANCES	FIRST APPEARANCE	YEAR
4	1576	Richard Grayson (New Earth)	\/wiki\/Richard_Grayson_(New_Earth)	Secret Identity	Good Characters	Blue Eyes	Black Hair	Male Characters	Heterosexual Characters	Living Characters	1237	April	1940

In [271]:
```
dc.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6896 entries, 0 to 6895
Data columns (total 13 columns):
page_id             6896 non-null int64
name                6896 non-null object
urlslug             6896 non-null object
ID                  6896 non-null object
ALIGN               6896 non-null object
EYE                 6896 non-null object
HAIR                6896 non-null object
SEX                 6896 non-null object
GSM                 6896 non-null object
ALIVE               6896 non-null object
APPEARANCES         6896 non-null int32
FIRST APPEARANCE    6896 non-null object
YEAR                6896 non-null int32
dtypes: int32(2), int64(1), object(10)
memory usage: 646.6+ KB
```

Кодирование категориальных признаков

In [272]:
```
binary_headers=[]
non_binary_headers=[]
[binary_headers.append(c) for c in cat_cols if dc[c].nunique()==2 ]
[non_binary_headers.append(c) for c in cat_cols if dc[c].nunique()!=2 and c not in ('name'
,'urlslug') ]
binary_columns=dc[binary_headers]
non_binary_columns=dc[non_binary_headers]
```

In [273]:
```
le = LabelEncoder()
cat_enc_le = le.fit_transform(non_binary_columns['ALIGN'])
#находит все уникальные значения и строит таблицу для соответствия каждой категории
#некоторому числу, затем преобразует значения в числа
non_binary_columns['ALIGN'].unique()
```

Out[273]:
```
array(['Good Characters', 'Bad Characters', 'Neutral Characters',
       'Reformed Criminals'], dtype=object)
```

In [274]:

```
#le.inverse_transform([0, 1, 2, 3])-обратная трансформация
np.unique(cat_enc_le)
```

Out[274]:

```
array([0, 1, 2, 3])
```

результат

In [275]:

```
### Так как подход так себе, обратимся к OneHot Encoding-y
non_binary_columns.head()
```

Out[275]:

	ID	ALIGN	EYE	HAIR	SEX	GSM	FIRST APPEARANCE
0	Secret Identity	Good Characters	Blue Eyes	Black Hair	Male Characters	Heterosexual Characters	May
1	Secret Identity	Good Characters	Blue Eyes	Black Hair	Male Characters	Heterosexual Characters	October
2	Secret Identity	Good Characters	Brown Eyes	Brown Hair	Male Characters	Heterosexual Characters	October
3	Public Identity	Good Characters	Brown Eyes	White Hair	Male Characters	Heterosexual Characters	February
4	Secret Identity	Good Characters	Blue Eyes	Black Hair	Male Characters	Heterosexual Characters	April

In [276]:

```
dc_cat = pd.get_dummies(dc[non_binary_headers])
dc_cat.head()
```

Out[276]:

	ID_Identity Unknown	ID_Public Identity	ID_Secret Identity	ALIGN_Bad Characters	ALIGN_God Characters	ALIGN_Neutral Characters	ALIGN_Reformed Criminals	EYE_Amber Eyes	EYE_Auburn Hair	EYE_Black Eyes	...	FIRST APPEARANCE_February	FIRST APPEARANCE_Holiday	FIRST APPEARANCE_January	FIRST APPEARANCE_July	FIRST APPEARANCE_June	FIRST APPEARANCE_March	FIRST APPEARANCE_May	FIRST APPEARANCE_November	FIRST APPEARANCE_October	FIRST APPEARANCE_September
0	0	0	1	0	1	0	0	0	0	0	...	0	0	0	0	0	0	1	0	0	0
1	0	0	1	0	1	0	0	0	0	0	...	0	0	0	0	0	0	0	0	1	0
2	0	0	1	0	1	0	0	0	0	0	...	0	0	0	0	0	0	0	0	1	0

	ID_Identity Unknown	ID_Public Identity	ID_Secret Identity	ALIGN_Bad Characters	ALIGN_Good Characters	ALIGN_Neutral Characters	ALIGN_Reformed Criminals	EYE_Amber Eyes	EYE_Auburn Hair	EYE_Black Eyes	...	FIRST APPEARANCE_February	FIRST APPEARANCE_Holiday	FIRST APPEARANCE_January	FIRST APPEARANCE_July	FIRST APPEARANCE_June	FIRST APPEARANCE_March	FIRST APPEARANCE_May	FIRST APPEARANCE_November	FIRST APPEARANCE_October	FIRST APPEARANCE_September
3	0	1	0	0	1	0	0	0	0	0	...	1	0	0	0	0	0	0	0	0	0
4	0	0	1	0	1	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

5 rows × 62 columns

In [277]:

```
dc.info()
num_cols
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6896 entries, 0 to 6895
Data columns (total 13 columns):
page_id             6896 non-null int64
name                6896 non-null object
urlslug             6896 non-null object
ID                  6896 non-null object
ALIGN               6896 non-null object
EYE                 6896 non-null object
HAIR                6896 non-null object
SEX                 6896 non-null object
GSM                 6896 non-null object
ALIVE               6896 non-null object
APPEARANCES         6896 non-null int32
FIRST APPEARANCE    6896 non-null object
YEAR                6896 non-null int32
dtypes: int32(2), int64(1), object(10)
memory usage: 646.6+ KB
```

Out[277]:

```
['page_id', 'APPEARANCES', 'YEAR']
```

Масштабирование данных

In [278]:

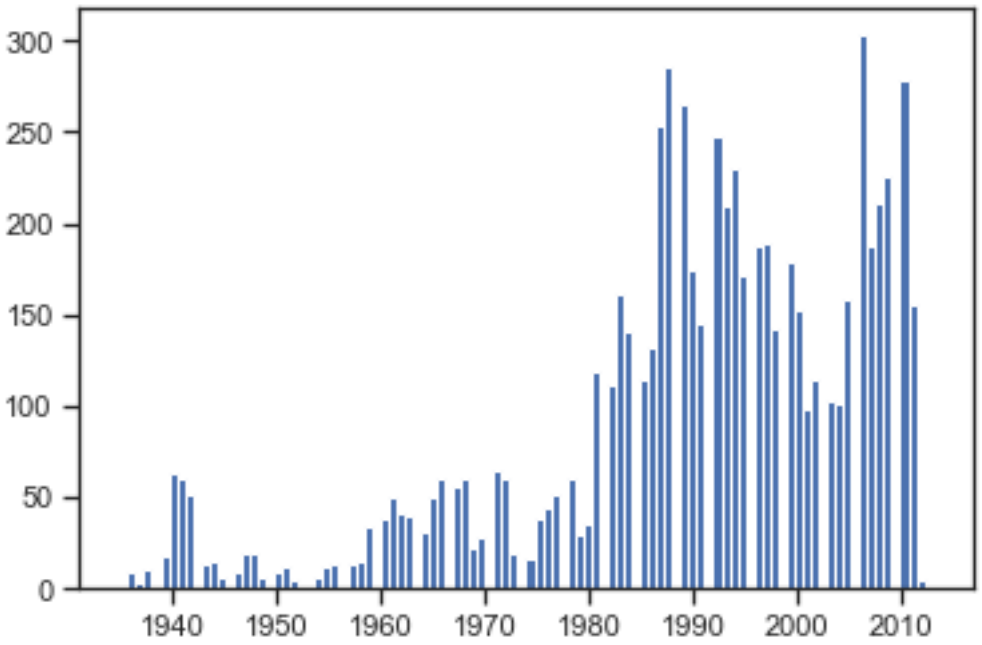
```
sc1 = MinMaxScaler()
dc1=[sc1.fit_transform(dc[[c]]) for c in ('APPEARANCES', 'YEAR')]
```

```
C:\Users\kotsi\Anaconda37\lib\site-packages\sklearn\preprocessing\data.py:323: DataConversionWarning: Data with input dtype int32 were all converted to float64 by MinMaxScaler.
```

```
  return self.partial_fit(X, y)
C:\Users\kotsi\Anaconda37\lib\site-packages\sklearn\preprocessing\data.py:323: DataConvers
ionWarning: Data with input dtype int32 were all converted to float64 by MinMaxScaler.
  return self.partial_fit(X, y)
```

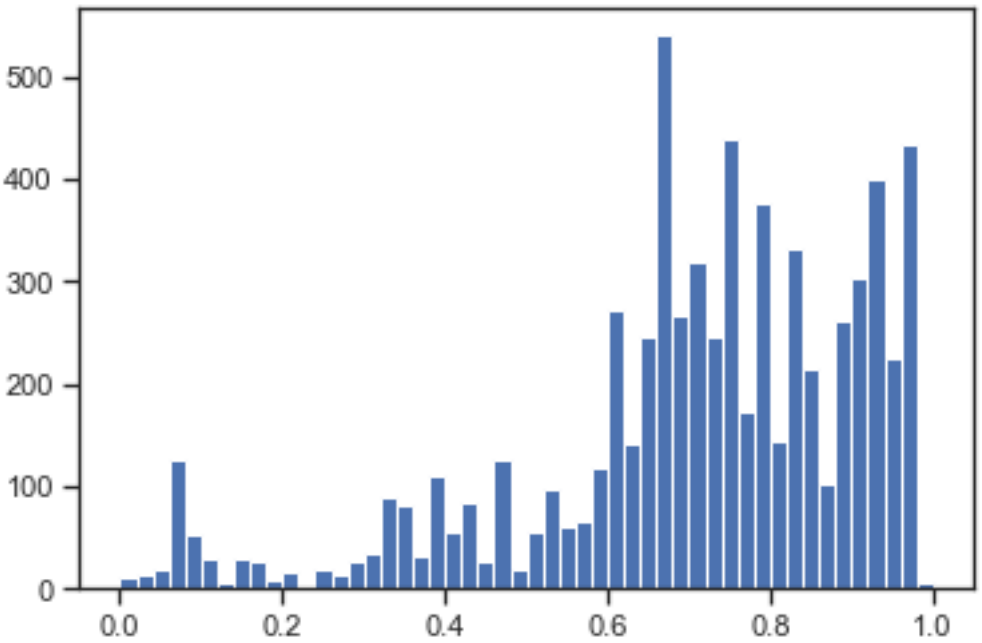
In [279]:

```
plt.hist(dc['YEAR'], 100)
plt.show()
```



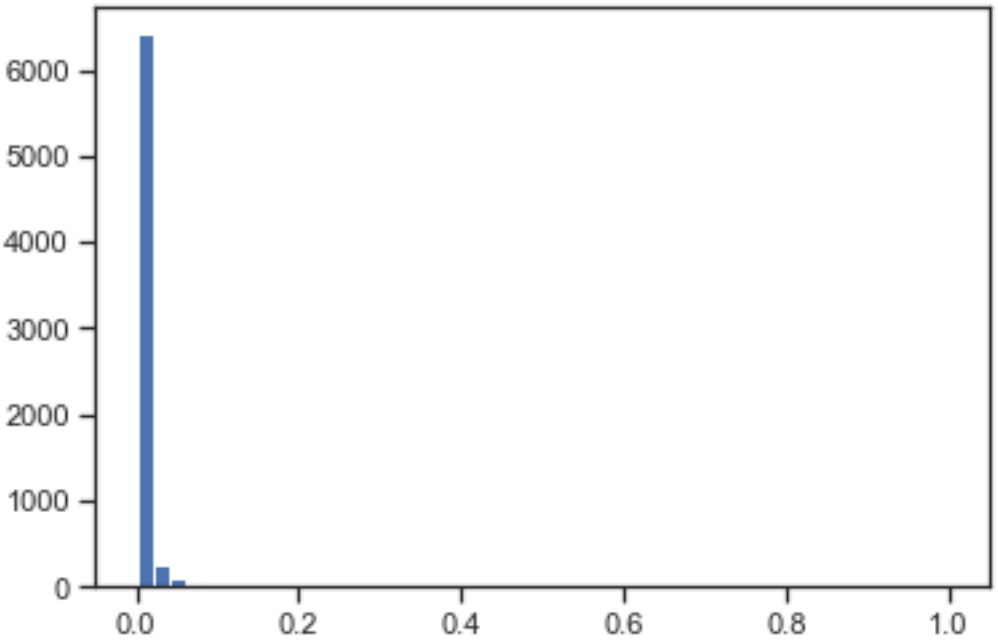
In [280]:

```
plt.hist(dc1[1], 50)
plt.show()
```



In [281]:

```
plt.hist(dc1[0], 50)
plt.show()
```



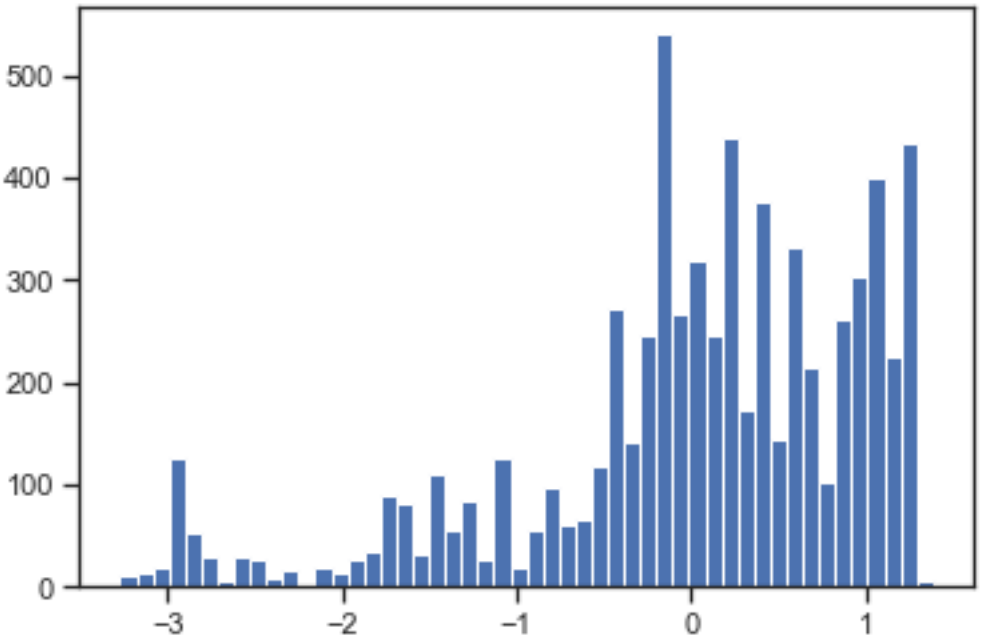
In [282]:

```
sc2 = StandardScaler()
sc2_data = sc2.fit_transform(dc[['YEAR']])
```

```
C:\Users\kotsi\Anaconda37\lib\site-packages\sklearn\preprocessing\data.py:625: DataConversionWarning: Data with input dtype int32 were all converted to float64 by StandardScaler.
  return self.partial_fit(X, y)
C:\Users\kotsi\Anaconda37\lib\site-packages\sklearn\base.py:462: DataConversionWarning: Data with input dtype int32 were all converted to float64 by StandardScaler.
  return self.fit(X, **fit_params).transform(X)
```

In [283]:

```
plt.hist(sc2_data, 50)
plt.show()
```



In [284]:

```
sc3 = Normalizer()
```

```
dc2= sc3.fit_transform(dc[['APPEARANCES', 'YEAR']])
```

In [285]:

```
dc2=pd.DataFrame(dc2)
```

In [286]:

```
dc2.columns=['APPEARANCES', 'YEAR']
```

In [287]:

```
dc2.head()
```

Out[287]:

	APPEARANCES	YEAR
0	0.847274	0.531156
1	0.782518	0.622628
2	0.624160	0.781297
3	0.552180	0.833725
4	0.537635	0.843178

In [288]:

```
dc2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6896 entries, 0 to 6895
Data columns (total 2 columns):
APPEARANCES    6896 non-null float64
YEAR           6896 non-null float64
dtypes: float64(2)
memory usage: 107.8 KB
```

In [289]:

```
DC_BIG = pd.concat([dc_cat,dc2], axis=1)
```

In [290]:

```
DC_BIG.head()
```

Out[290]:

	ID_Identity Unknown	ID_Public Identity	ID_Secret Identity	ALIGN_Bad Characters	ALIGN_Good Characters	ALIGN_Neutral Characters	ALIGN_Reformed Criminals	EYE_Amber Eyes	EYE_Auburn Hair	EYE_Black Eyes	...	FIRST APPEARANCE_January	FIRST APPEARANCE_July	FIRST APPEARANCE_June	FIRST APPEARANCE_March	FIRST APPEARANCE_May	FIRST APPEARANCE_November	FIRST APPEARANCE_October	FIRST APPEARANCE_September	APPEARANCES	YEAR
0	0	0	1	0	1	0	0	0	0	0	...	0	0	0	0	1	0	0	0	0.847274	0.531156
1	0	0	1	0	1	0	0	0	0	0	...	0	0	0	0	0	0	1	0	0.782518	0.622628
2	0	0	1	0	1	0	0	0	0	0	...	0	0	0	0	0	0	1	0	0.624160	0.781297
3	0	1	0	0	1	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0.552180	0.833725
4	0	0	1	0	1	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0.537635	0.843178

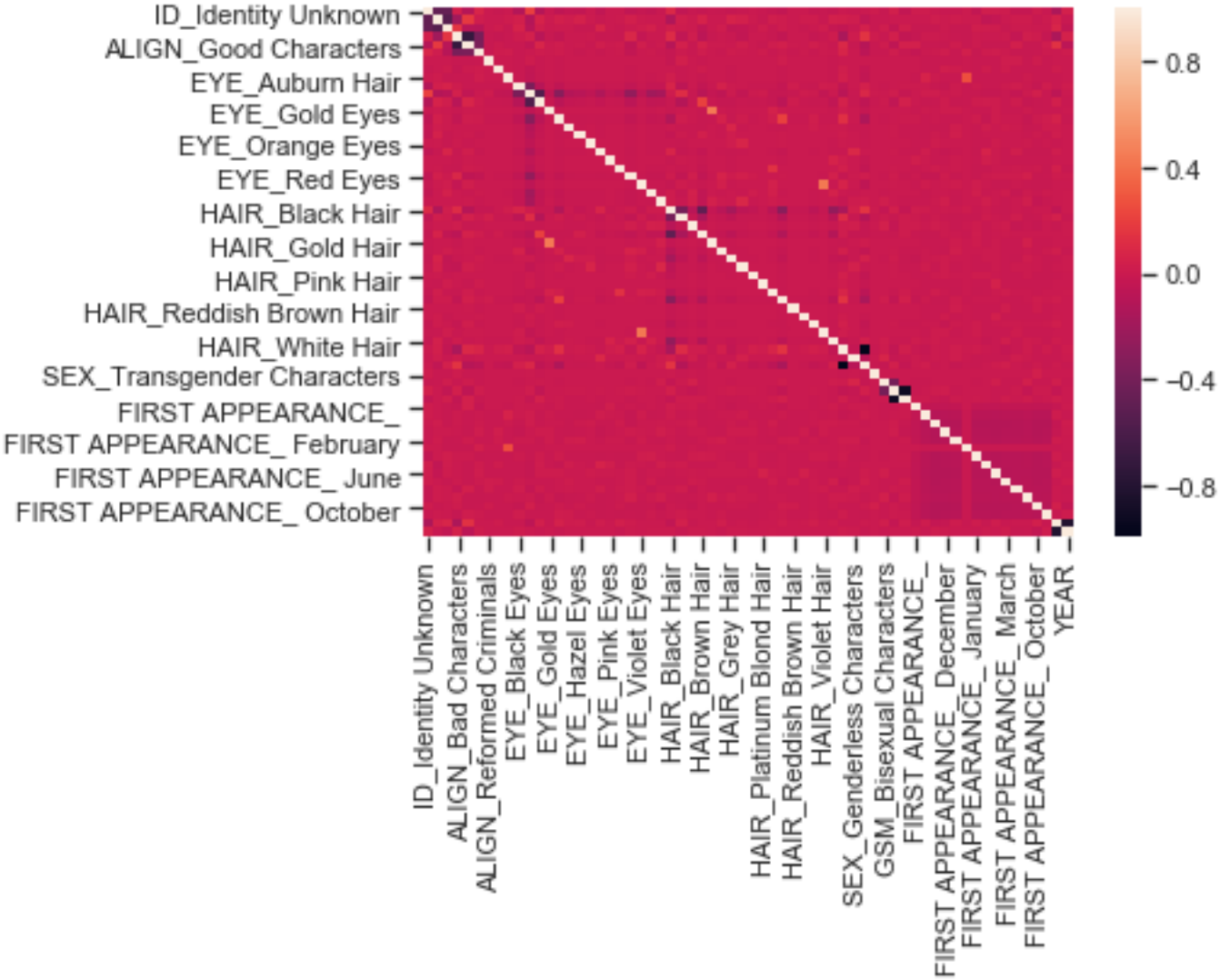
5 rows × 64 columns

In [291]:

```
sns.heatmap(DC_BIG.corr())
```

Out[291]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1bb81ace390>
```



In [292]:

```
print(DC_BIG.corr()['APPEARANCES'].abs().sort_values(ascending=False).head(10))
```

```
APPEARANCES                1.000000
YEAR                       0.818192
ALIGN_Good Characters      0.141453
ALIGN_Bad Characters       0.140337
ID_Identity Unknown        0.137600
ID_Secret Identity         0.080220
EYE_Blue Eyes              0.062221
ID_Public Identity         0.050905
EYE_Green Eyes             0.050818
HAIR_Black Hair            0.050293
Name: APPEARANCES, dtype: float64
```