DataAnalysis_old_dataset

June 11, 2023

[ ]:

```
# Define where you are running the code: colab or local
RunInColab          = True     # (False: no  | True: yes)

# If running in colab:
if RunInColab:
    # Mount your google drive in google colab
    from google.colab import drive
    drive.mount('/content/drive')

    # Find location
    #!pwd
    #!ls
    #!ls "/content/drive/My Drive/Colab Notebooks/MachineLearningWithPython/"

    # Define path del proyecto
    Ruta            = "/content/drive/My Drive/Colab Notebooks/"

else:
    # Define path del proyecto
    Ruta            = ""
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
```

[ ]:

```
# Import the packages that we will be using
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Dataset url
url = Ruta + "data_filtered.csv"

# Load the dataset
dataset = pd.read_csv(url)
```

[ ]:

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

[ ]:
```
import time
start = time.time()
```

[ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pickle
from sklearn.metrics import confusion_matrix, precision_score,␣
 ↪classification_report
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import StandardScaler,LabelEncoder,OneHotEncoder
from sklearn.model_selection import␣
 ↪cross_val_score,StratifiedKFold,RandomizedSearchCV,cross_val_predict
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier, RandomForestRegressor
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import␣
 ↪confusion_matrix,precision_score,f1_score,recall_score
from sklearn.neural_network import MLPClassifier, MLPRegressor
plt.style.use('seaborn')

np.set_printoptions(precision=4)
```

0.1 Read data

[ ]:
```
data = dataset
data['date'] = pd.to_datetime(data['date'])
```

[ ]:
```
len(data)
```

[ ]:
```
5583
```

[ ]:
```
data.head()
```

[ ]:
```
   year       date fp1_date fp1_time fp2_date fp2_time fp3_date fp3_time  \
0  2011 2011-03-27       \N       \N       \N       \N       \N       \N
1  2010 2010-03-28       \N       \N       \N       \N       \N       \N
2  2017 2017-03-26       \N       \N       \N       \N       \N       \N
3  2018 2018-03-25       \N       \N       \N       \N       \N       \N
4  2019 2019-03-17       \N       \N       \N       \N       \N       \N

  quali_date quali_time  … constructor constructor_nationality  \
0         \N         \N  …   Alpine F1                     Fre
```

```
1         \N         \N  …   Alpine F1                     Fre
2         \N         \N  …   Alpine F1                     Fre
3         \N         \N  …   Alpine F1                     Fre
4         \N         \N  …   Alpine F1                     Fre

                          GP_name  country            driver age_at_gp_in_days  \
0  Albert Park Grand Prix Circuit      Aus     Nick Heidfeld             12374
1  Albert Park Grand Prix Circuit      Aus     Robert Kubica              9242
2  Albert Park Grand Prix Circuit      Aus   Nico Hülkenberg             10812
3  Albert Park Grand Prix Circuit      Aus   Nico Hülkenberg             11176
4  Albert Park Grand Prix Circuit      Aus   Nico Hülkenberg             11533

  driver_home  constructor_home  driver_dnf  constructor_dnf
0           0                 0           0                1
1           0                 0           0                0
2           0                 0           0                1
3           0                 0           0                0
4           0                 0           0                1

[5 rows x 27 columns]
```

[ ]:
```
new_test = data["driver_dnf"]
new_test.head
```

[ ]:
```
<bound method NDFrame.head of 0       0
1       0
2       0
3       0
4       0
       ..
5578    0
5579    0
5580    0
5581    0
5582    0
Name: driver_dnf, Length: 5583, dtype: int64>
```

[ ]:
```
data['year'].unique()
```

[ ]:
```
array([2011, 2010, 2017, 2018, 2019, 2016, 2014, 2015, 2012, 2013, 2022,
       2023, 2020, 2021])
```

Si utilizan datos del 2023, por que al entrenar los modelos no usan el archivo data_filtered_2021.csv, usan el data_filtered.csv

0.2 Do data transformations

El calculo de driver_confidence y constructor reliability es incorrecto, ya que el calculo de estas variables es de todo el historial de datos que hay. Y esto nos da dos sesgos, uno el modelo tiene conocimiento del presente pasado y futuro, y el segundo es que tan bueno es una escuderia o un corredor puede cambiar cada 2 o 3 años, aunque hay algunos que duran mas. Pero en si seria bueno limitarlo a 2 años antes de la carrera estos datos.

[ ]:
```python
def get_reliability_driver_const(data, column):
    # Solo vamos a calcular 2 año atras la informacion de cuantas carrreras entro y cuantas veces tuvo dnf, en vez de calcular de todo el historial de las carreras
    data_temp_dnf = data.copy()
    data_temp_dnf['index_column'] = data_temp_dnf.index
    data_temp_dnf.set_index('date', inplace=True)
    data_temp_dnf.sort_index(inplace=True)


    window_size = '730D' # 2 years

    # data_temp_dnf['rolling_sum_1'] = data_temp_dnf.groupby('driver')['driver_dnf'].apply(lambda x: x.shift().rolling(window_size).sum())
    # data_temp_dnf['rolling_sum_2'] = data_temp_dnf.groupby('driver')['driver_dnf'].apply(lambda x: x.shift().rolling(window_size).count())

    all_columns = data[column].unique()

    for column_row in all_columns:
        data_temp_dnf.loc[data_temp_dnf[column] == column_row, 'total_dnf'] = data_temp_dnf[data_temp_dnf[column] == column_row].shift().rolling(window_size)[f'{column}_dnf'].sum()
        data_temp_dnf.loc[data_temp_dnf[column] == column_row, 'total_races'] = data_temp_dnf[data_temp_dnf[column] == column_row].shift().rolling(window_size)[f'{column}_dnf'].count()


    data_temp_dnf['dnf_ratio'] = 1 - (data_temp_dnf['total_dnf'] / data_temp_dnf['total_races'])

    data_temp_dnf.reset_index(inplace=True)
    data_temp_dnf.set_index('index_column', inplace=True)
    data_temp_dnf.sort_index(inplace=True)
    data_temp_dnf.reset_index(inplace=True)

    return data_temp_dnf['dnf_ratio']
```

[ ]:

```
# dnf_by_driver = data.groupby('driver').sum()['driver_dnf']
# driver_race_entered = data.groupby('driver').count()['driver_dnf']
# driver_dnf_ratio = (dnf_by_driver/driver_race_entered)
# driver_confidence = 1-driver_dnf_ratio
# driver_confidence_dict = dict(zip(driver_confidence.index,driver_confidence))
```

[ ]:

```
# dnf_by_constructor = data.groupby('constructor').sum()['constructor_dnf']
# constructor_race_entered = data.groupby('constructor').
 ↪count()['constructor_dnf']
# constructor_dnf_ratio = (dnf_by_constructor/constructor_race_entered)
# constructor_reliability = 1-constructor_dnf_ratio
# constructor_reliability_dict = dict(zip(constructor_reliability.
 ↪index,constructor_reliability))
```

[ ]:

```
# data['driver_confidence'] = data['driver'].apply(lambda x:
 ↪driver_confidence_dict[x])
# data['constructor_reliability'] = data['constructor'].apply(lambda x:
 ↪constructor_reliability_dict[x])

data['driver_confidence'] = get_reliability_driver_const(data, 'driver')
data['driver_confidence'] = data['driver_confidence'].fillna(0)
data['constructor_reliability'] = get_reliability_driver_const(data,␣
 ↪'constructor')
data['constructor_reliability'] = data['constructor_reliability'].fillna(0)


#removing retired drivers and constructors
active_constructors = ['Alpine F1', 'Williams', 'McLaren', 'Ferrari',␣
 ↪'Mercedes',
                       'AlphaTauri', 'Aston Martin', 'Alfa Romeo', 'Red Bull',
                       'Haas F1 Team']
active_drivers = ['Daniel Ricciardo', 'Mick Schumacher', 'Carlos Sainz',
                  'Valtteri Bottas', 'Lance Stroll', 'George Russell',
                  'Lando Norris', 'Sebastian Vettel', 'Kimi Räikkönen',
                  'Charles Leclerc', 'Lewis Hamilton', 'Yuki Tsunoda',
                  'Max Verstappen', 'Pierre Gasly', 'Fernando Alonso',
                  'Sergio Pérez', 'Esteban Ocon', 'Antonio Giovinazzi',
                  'Nikita Mazepin','Nicholas Latifi']
data['active_driver'] = data['driver'].apply(lambda x: int(x in active_drivers))
data['active_constructor'] = data['constructor'].apply(lambda x: int(x in␣
 ↪active_constructors))
```

La variable de dnf no se usaria para entrenar el modelo directamente, porque esta ya nos da la inforacion de si acabo la carrera o no

[ ]:

```
import os
if not os.path.exists('./models'):
    os.mkdir('./models')
```

```
[ ]: data.dtypes
```

```
[ ]: year                                int64
     date                       datetime64[ns]
     fp1_date                           object
     fp1_time                           object
     fp2_date                           object
     fp2_time                           object
     fp3_date                           object
     fp3_time                           object
     quali_date                         object
     quali_time                         object
     sprint_date                        object
     sprint_time                        object
     quali_pos                           int64
     statusId                            int64
     position                            int64
     dob                                object
     driver_nationality                 object
     constructor                        object
     constructor_nationality            object
     GP_name                            object
     country                            object
     driver                             object
     age_at_gp_in_days                   int64
     driver_home                         int64
     constructor_home                    int64
     driver_dnf                          int64
     constructor_dnf                     int64
     driver_confidence                 float64
     constructor_reliability           float64
     active_driver                       int64
     active_constructor                  int64
     dtype: object
```

0.3 Study the positions (y) and qualification position variables

```
[ ]: def position_index(x):
         if x < 4:
             return 0 # 1
         if x > 10:
             return 2 # 3
         else:
             return 1 # 2
```

[ ]:

```
# pearson’, ‘kendall’, ‘spearman
def showCorrelation(data, nCols):
    methods=["pearson", "kendall", "spearman"]

    fig, axes = plt.subplots(nrows=len(methods), ncols=nCols, figsize=(5 *␣
 ↪len(methods), 10))

    for i, method in enumerate(methods):
        ax = axes[i]
        corr = data.corr(method=method)
        sns.heatmap(corr, annot=True, ax=ax)
        ax.set_title(f'{method.capitalize()} Correlation')

    plt.tight_layout()
    plt.show()
```

[ ]:

```
def factorizeColumns(data, colName, factColName):
    data[factColName] = pd.factorize(data[colName])[0]
    data[factColName].head
```

[ ]:

```
def graph_diff_pos_quali_pos(all_gps_year, all_year_data):
  ncols = 2
  nrows = (len(all_gps_year) + 1) // ncols

  fig, axes = plt.subplots(nrows, ncols, figsize=(20, 70))

  for index, race in enumerate(all_gps_year.iterrows()):
      race = race[1]
      ax = axes[index // ncols, index % ncols]

      race_name = race['GP_name']
      race_date = race['date']
      race_data = all_year_data[all_year_data['GP_name'] == race_name]

      unique_race = race_data[race_data['date'] == race_date]

      ax.scatter(unique_race["driver"], unique_race["quali_pos"], color='k')
      ax.scatter(unique_race["driver"], unique_race["position"], color='g')

      ax.set_xticklabels(unique_race["driver"], rotation=90)
      ax.set_title(f"Grand prix {race_name}, {race_date}")

  plt.tight_layout()
  plt.show()
```

[ ]:

```
data["active_driver"].head()
```

```
[ ]: 0    0
     1    0
     2    0
     3    0
     4    0
     Name: active_driver, dtype: int64
```

```
[ ]: temp = data["driver"].unique()
     len(temp)
```

```
[ ]: 76
```

```
[ ]: year2020 = data[data["year"] == 2020]
     races = year2020["GP_name"].unique()
```

```
[ ]: all_gps_2020 = year2020[['GP_name', 'date']].drop_duplicates()
```

```
[ ]: race_data_temp = year2020[year2020['GP_name'] == races[0]]

     race_data_temp[['position', 'quali_pos', 'driver', 'constructor', 'date']]
```

```
[ ]:      position  quali_pos              driver   constructor       date
     655         6          6    Daniel Ricciardo     Alpine F1 2020-11-29
     656         7          7    Daniel Ricciardo     Alpine F1 2020-12-06
     661         7          7        Esteban Ocon     Alpine F1 2020-11-29
     662        11         11        Esteban Ocon     Alpine F1 2020-12-06
     683        14         14      George Russell      Williams 2020-11-29
     685        20         20     Nicholas Latifi      Williams 2020-11-29
     686        17         16     Nicholas Latifi      Williams 2020-12-06
     689        18         17         Jack Aitken      Williams 2020-12-06
     706        15         15        Carlos Sainz       McLaren 2020-11-29
     707         8          8        Carlos Sainz       McLaren 2020-12-06
     712         9          9        Lando Norris       McLaren 2020-11-29
     713        15         19        Lando Norris       McLaren 2020-12-06
     727        11         11    Sebastian Vettel       Ferrari 2020-11-29
     728        13         13    Sebastian Vettel       Ferrari 2020-12-06
     741        12         12     Charles Leclerc       Ferrari 2020-11-29
     742         4          4     Charles Leclerc       Ferrari 2020-12-06
     781         1          1      Lewis Hamilton      Mercedes 2020-11-29
     790         2          2     Valtteri Bottas      Mercedes 2020-11-29
     791         1          1     Valtteri Bottas      Mercedes 2020-12-06
     793         2          2      George Russell      Mercedes 2020-12-06
     866         3          3      Max Verstappen      Red Bull 2020-11-29
     867         3          3      Max Verstappen      Red Bull 2020-12-06
     872         4          4     Alexander Albon      Red Bull 2020-11-29
     873        12         12     Alexander Albon      Red Bull 2020-12-06
     875        17         17      Kimi Räikkönen    Alfa Romeo 2020-11-29
```

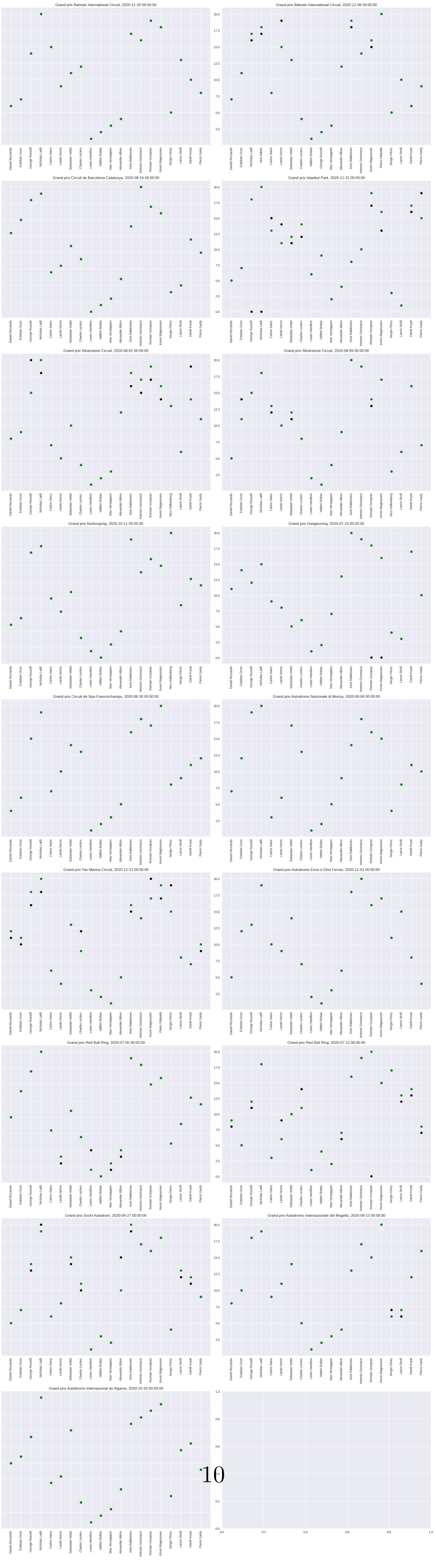
```
876        19         18      Kimi Räikkönen    Alfa Romeo 2020-12-06
881        16         16  Antonio Giovinazzi    Alfa Romeo 2020-11-29
882        14         14  Antonio Giovinazzi    Alfa Romeo 2020-12-06
904        19         19     Romain Grosjean  Haas F1 Team 2020-11-29
910        18         18     Kevin Magnussen  Haas F1 Team 2020-11-29
911        16         15     Kevin Magnussen  Haas F1 Team 2020-12-06
914        20         20   Pietro Fittipaldi  Haas F1 Team 2020-12-06
923         5          5        Sergio Pérez  Aston Martin 2020-11-29
924         5          5        Sergio Pérez  Aston Martin 2020-12-06
926        13         13        Lance Stroll  Aston Martin 2020-11-29
927        10         10        Lance Stroll  Aston Martin 2020-12-06
928        10         10        Daniil Kvyat    AlphaTauri 2020-11-29
929         6          6        Daniil Kvyat    AlphaTauri 2020-12-06
930         8          8        Pierre Gasly    AlphaTauri 2020-11-29
931         9          9        Pierre Gasly    AlphaTauri 2020-12-06
```

```
[ ]: print(all_gps_2020)
```

```
                                   GP_name       date
655          Bahrain International Circuit 2020-11-29
656          Bahrain International Circuit 2020-12-06
949         Circuit de Barcelona-Catalunya 2020-08-16
1479                         Istanbul Park 2020-11-15
1570                   Silverstone Circuit 2020-08-02
1571                   Silverstone Circuit 2020-08-09
1861                           Nürburgring 2020-10-11
1933                           Hungaroring 2020-07-19
2282          Circuit de Spa-Francorchamps 2020-08-30
2556          Autodromo Nazionale di Monza 2020-09-06
3560                    Yas Marina Circuit 2020-12-13
4193         Autodromo Enzo e Dino Ferrari 2020-11-01
4257                         Red Bull Ring 2020-07-05
4258                         Red Bull Ring 2020-07-12
4574                        Sochi Autodrom 2020-09-27
5363  Autodromo Internazionale del Mugello 2020-09-13
5383    Autódromo Internacional do Algarve 2020-10-25
```

0.3.1 Graph the difference in qualification position and position in the year 2020

```
[ ]: graph_diff_pos_quali_pos(all_gps_2020, year2020)
```



Grand prix Bahrain International Circuit, 2020-11-29 00:00:00

Grand prix Bahrain International Circuit, 2020-12-06 00:00:00

Grand prix Circuit de Barcelona-Catalunya, 2020-08-16 00:00:00

Grand prix Istanbul Park, 2020-11-15 00:00:00

Grand prix Silverstone Circuit, 2020-08-02 00:00:00

Grand prix Silverstone Circuit, 2020-08-09 00:00:00

Grand prix Nürburgring, 2020-10-11 00:00:00

Grand prix Hungaroring, 2020-07-19 00:00:00

Grand prix Circuit de Spa-Francorchamps, 2020-08-30 00:00:00

Grand prix Autodromo Nazionale di Monza, 2020-09-06 00:00:00

Grand prix Yas Marina Circuit, 2020-12-13 00:00:00

Grand prix Autodromo Enzo e Dino Ferrari, 2020-11-01 00:00:00

Grand prix Red Bull Ring, 2020-07-05 00:00:00

Grand prix Red Bull Ring, 2020-07-12 00:00:00

Grand prix Sochi Autodrom, 2020-09-27 00:00:00

Grand prix Autodromo Internazionale del Mugello, 2020-09-13 00:00:00

Grand prix Autódromo Internacional do Algarve, 2020-10-25 00:00:00

[ ]:

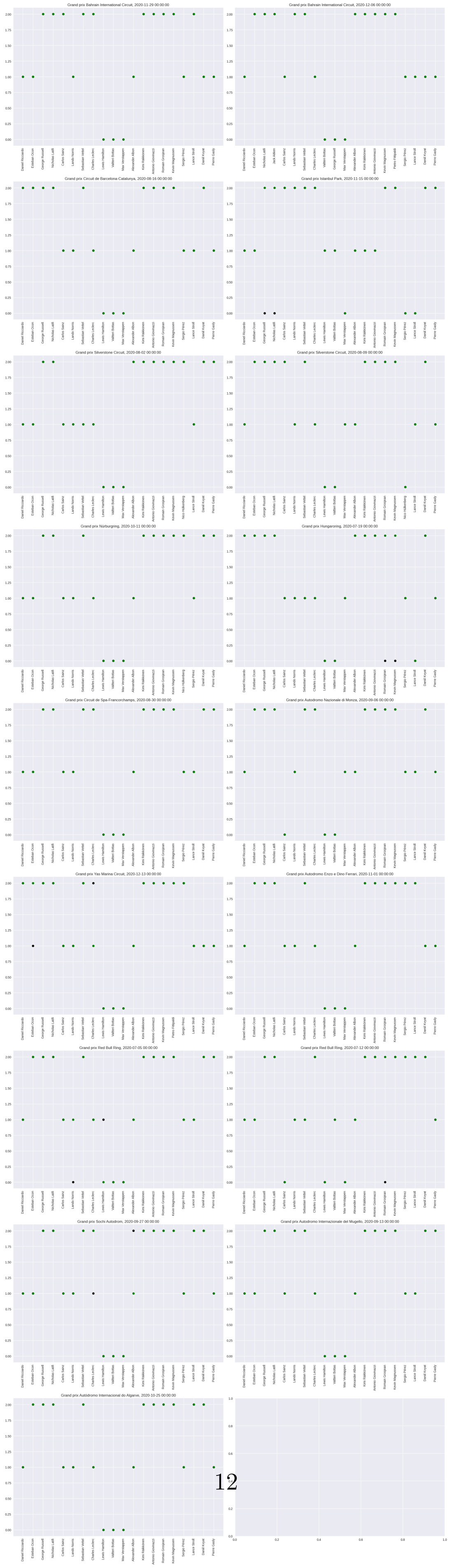
```
year2020_in_3_classes = year2020.copy()
year2020_in_3_classes["position"] = year2020_in_3_classes["position"].
 ↪apply(lambda x: position_index(x))
year2020_in_3_classes["quali_pos"] = year2020_in_3_classes["quali_pos"].
 ↪apply(lambda x: position_index(x))
```

[ ]:

```
all_gps_2020_in_3_classes = year2020_in_3_classes[['GP_name', 'date']].
 ↪drop_duplicates()
```

[ ]:

```
graph_diff_pos_quali_pos(all_gps_2020_in_3_classes, year2020_in_3_classes)
```


Grand prix Bahrain International Circuit, 2020-11-29 00:00:00

Grand prix Bahrain International Circuit, 2020-12-06 00:00:00

Grand prix Circuit de Barcelona-Catalunya, 2020-08-16 00:00:00

Grand prix Istanbul Park, 2020-11-15 00:00:00

Grand prix Silverstone Circuit, 2020-08-02 00:00:00

Grand prix Silverstone Circuit, 2020-08-09 00:00:00

Grand prix Nürburgring, 2020-10-11 00:00:00

Grand prix Hungaroring, 2020-07-19 00:00:00

Grand prix Circuit de Spa-Francorchamps, 2020-08-30 00:00:00

Grand prix Autodromo Nazionale di Monza, 2020-09-06 00:00:00

Grand prix Yas Marina Circuit, 2020-12-13 00:00:00

Grand prix Autodromo Enzo e Dino Ferrari, 2020-11-01 00:00:00

Grand prix Red Bull Ring, 2020-07-05 00:00:00

Grand prix Red Bull Ring, 2020-07-12 00:00:00

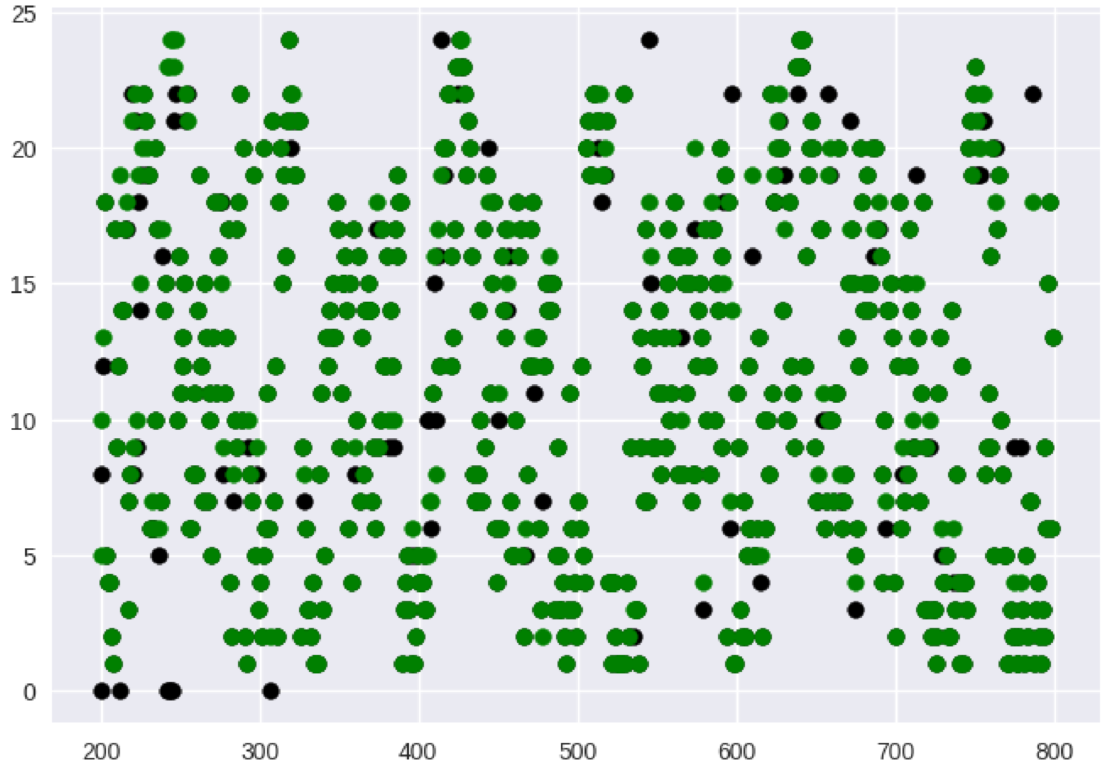
Grand prix Sochi Autodrom, 2020-09-27 00:00:00

Grand prix Autodromo Internazionale del Mugello, 2020-09-13 00:00:00

Grand prix Autódromo Internacional do Algarve, 2020-10-25 00:00:00

[ ]:

```
fig, ax = plt.subplots()

#ax.plot(list(data.index.values.tolist())[:50], data["quali_pos"][:50], 'o',␣
 ↪label='Graph 1')
#ax.plot(list(data.index.values.tolist())[:50], data["position"][:50], 'x',␣
 ↪label='Graph 2')
#ax.set_xlabel('ID')
#ax.set_ylabel('Position')
#ax.set_title('Plot of X vs Y')
#ax.grid(True)

#ax.legend()
#plt.show()

plt.scatter(list(data.index.values.tolist())[200:800], data["quali_pos"][200:
 ↪800], color='k')
plt.scatter(list(data.index.values.tolist())[200:800], data["position"][200:
 ↪800],color='g')
plt.show()
```



0.3.2 Calculate the mean difference and standard deviation of changes in qualification position and position

```
[ ]: diff_quali_pos = data[["position", "quali_pos"]]
     diff_quali_pos['position_3_classes'] = diff_quali_pos['position'].apply(lambda␣
      ↪x: position_index(x))
     diff_quali_pos['quali_pos_3_classes'] = diff_quali_pos['quali_pos'].
      ↪apply(lambda x: position_index(x))

     diff_quali_pos['diff'] = diff_quali_pos['quali_pos'] -␣
      ↪diff_quali_pos['position']
     diff_quali_pos['diff_3_classes'] = diff_quali_pos['quali_pos_3_classes'] -␣
      ↪diff_quali_pos['position_3_classes']
```

```
[ ]: print(f"Unique qualification position: {diff_quali_pos['quali_pos'].unique()}")
     print(f"Unqiue positions: {diff_quali_pos['position'].unique()}")
```

```
Unique qualification position: [18  9 11  7  6 12 14 13 19  8 17 20  3 15 16  0
4  2 10  1  5 23 21 24
 22]
Unqiue positions: [18  9 12  8 11  6 15 14 20 17  3  7 16 10 19  4  2 13  1  5
21 22 23 24]
```

Qualification position 0?

La informacion del dataset parece estar mal, la carrera del 2015 es un ejemplo en donde Valtteri Bottas no llego a competir en la carrera, por esto el quali_pos 0, pero la position en la carrera dice que es 6 cuando deberia de ser DNS (Did not start), y las posicion en esa carrera estan mal tambien ya que Vettel quedo en 3 lugar y esta dice en cuarto lugar

```
[ ]: data[data['quali_pos'] == 0][["date", "GP_name", "position", "quali_pos",␣
      ↪"driver"]]
```

```
[ ]:            date                          GP_name  position  quali_pos  \
     26   2015-03-15  Albert Park Grand Prix Circuit         6          0
     196  2023-04-02  Albert Park Grand Prix Circuit        20          0
     200  2017-03-26  Albert Park Grand Prix Circuit        10          0
     212  2023-04-02  Albert Park Grand Prix Circuit        19          0
     242  2011-03-27  Albert Park Grand Prix Circuit        23          0
     243  2012-03-18  Albert Park Grand Prix Circuit        23          0
     244  2011-03-27  Albert Park Grand Prix Circuit        24          0
     245  2012-03-18  Albert Park Grand Prix Circuit        24          0
     307  2017-10-01    Sepang International Circuit         2          0
     854  2021-03-28   Bahrain International Circuit        11          0
     945  2019-05-12  Circuit de Barcelona-Catalunya        16          0
     1362 2015-05-24               Circuit de Monaco         8          0
     1393 2011-05-29               Circuit de Monaco        10          0
     1401 2016-05-29               Circuit de Monaco        22          0
```

```
1421 2016-05-29               Circuit de Monaco        21          0
1485 2020-11-15                   Istanbul Park        18          0
1487 2020-11-15                   Istanbul Park        20          0
1730 2018-07-08             Silverstone Circuit        20          0
1776 2021-07-18             Silverstone Circuit         5          0
2153 2021-08-01                     Hungaroring        14          0
2173 2020-07-19                     Hungaroring        18          0
2178 2020-07-19                     Hungaroring        16          0
2196 2022-07-31                     Hungaroring        19          0
2290 2019-09-01    Circuit de Spa-Francorchamps        20          0
2477 2021-08-29    Circuit de Spa-Francorchamps         7          0
2496 2021-08-29    Circuit de Spa-Francorchamps        19          0
2543 2022-08-28    Circuit de Spa-Francorchamps        12          0
2545 2022-08-28    Circuit de Spa-Francorchamps        19          0
2770 2019-09-08    Autodromo Nazionale di Monza        10          0
2817 2021-09-12    Autodromo Nazionale di Monza         6          0
2819 2021-09-12    Autodromo Nazionale di Monza        17          0
2941 2022-10-02       Marina Bay Street Circuit        11          0
3076 2019-10-13                  Suzuka Circuit        20          0
3509 2021-11-14      Autódromo José Carlos Pace        14          0
3549 2022-11-13      Autódromo José Carlos Pace        19          0
4044 2019-06-09       Circuit Gilles Villeneuve        10          0
4217 2021-04-18   Autodromo Enzo e Dino Ferrari        13          0
4238 2022-04-24   Autodromo Enzo e Dino Ferrari        14          0
4268 2016-07-03                   Red Bull Ring        10          0
4277 2019-06-30                   Red Bull Ring        19          0
4369 2016-07-03                   Red Bull Ring        20          0
4426 2022-07-10                   Red Bull Ring        12          0
4444 2020-07-12                   Red Bull Ring        20          0
4694 2019-09-29                  Sochi Autodrom        19          0
4737 2019-04-28               Baku City Circuit        18          0
4783 2023-04-30               Baku City Circuit        12          0
4838 2019-04-28               Baku City Circuit        20          0
4839 2019-04-28               Baku City Circuit        19          0
4851 2023-04-30               Baku City Circuit        17          0
4946 2022-10-23         Circuit of the Americas        18          0
5067 2019-11-03         Circuit of the Americas        19          0
5291 2021-06-20             Circuit Paul Ricard        20          0
5475 2022-03-27         Jeddah Corniche Circuit        14          0
5500 2022-05-08   Miami International Autodrome        13          0
5501 2022-05-08   Miami International Autodrome        10          0
5525 2021-09-05          Circuit Park Zandvoort        14          0
5547 2021-09-05          Circuit Park Zandvoort        16          0

                  driver
26       Valtteri Bottas
196         Sergio Pérez
```

```
200    Daniel Ricciardo
212     Valtteri Bottas
242   Vitantonio Liuzzi
243    Pedro de la Rosa
244  Narain Karthikeyan
245  Narain Karthikeyan
307       Kimi Räikkönen
854         Sergio Pérez
945      Nico Hülkenberg
1362        Carlos Sainz
1393        Sergio Pérez
1401         Felipe Nasr
1421      Max Verstappen
1485      George Russell
1487     Nicholas Latifi
1730     Brendon Hartley
1776        Sergio Pérez
2153  Antonio Giovinazzi
2173     Romain Grosjean
2178     Kevin Magnussen
2196        Pierre Gasly
2290       Robert Kubica
2477        Sergio Pérez
2496      Kimi Räikkönen
2543        Pierre Gasly
2545        Yuki Tsunoda
2770      Kimi Räikkönen
2817        Pierre Gasly
2819        Yuki Tsunoda
2941      George Russell
3076       Robert Kubica
3509      Kimi Räikkönen
3549        Yuki Tsunoda
4044     Kevin Magnussen
4217    Sebastian Vettel
4238         Guanyu Zhou
4268        Felipe Massa
4277      George Russell
4369        Daniil Kvyat
4426     Valtteri Bottas
4444     Romain Grosjean
4694     Alexander Albon
4737       Robert Kubica
4783        Esteban Ocon
4838        Pierre Gasly
4839      Kimi Räikkönen
4851     Nico Hülkenberg
```

```
4946        Esteban Ocon
5067        Sergio Pérez
5291         Yuki Tsunoda
5475     Mick Schumacher
5500    Sebastian Vettel
5501        Lance Stroll
5525     Nicholas Latifi
5547        Sergio Pérez
```

```
[ ]: data[data['date'] == '2015-03-15'][['quali_pos', 'position', 'GP_name', 'date',␣
      ↪'driver']]
```

```
[ ]:      quali_pos  position                          GP_name        date  \
     16           3         3  Albert Park Grand Prix Circuit 2015-03-15
     26           0         6  Albert Park Grand Prix Circuit 2015-03-15
     41          16        17  Albert Park Grand Prix Circuit 2015-03-15
     52          17        18  Albert Park Grand Prix Circuit 2015-03-15
     65           4         4  Albert Park Grand Prix Circuit 2015-03-15
     71           5         5  Albert Park Grand Prix Circuit 2015-03-15
     111          2         2  Albert Park Grand Prix Circuit 2015-03-15
     115          1         1  Albert Park Grand Prix Circuit 2015-03-15
     142          7         8  Albert Park Grand Prix Circuit 2015-03-15
     145         11        12  Albert Park Grand Prix Circuit 2015-03-15
     156         13        14  Albert Park Grand Prix Circuit 2015-03-15
     162         14        15  Albert Park Grand Prix Circuit 2015-03-15
     178         15        16  Albert Park Grand Prix Circuit 2015-03-15
     182         10        11  Albert Park Grand Prix Circuit 2015-03-15
     198          6         7  Albert Park Grand Prix Circuit 2015-03-15
     202         12        13  Albert Park Grand Prix Circuit 2015-03-15
     221          8         9  Albert Park Grand Prix Circuit 2015-03-15
     223          9        10  Albert Park Grand Prix Circuit 2015-03-15

                    driver
     16       Felipe Massa
     26    Valtteri Bottas
     41      Jenson Button
     52    Kevin Magnussen
     65   Sebastian Vettel
     71     Kimi Räikkönen
     111       Nico Rosberg
     115     Lewis Hamilton
     142       Carlos Sainz
     145     Max Verstappen
     156    Nico Hülkenberg
     162       Sergio Pérez
     178    Marcus Ericsson
     182        Felipe Nasr
```

```
198   Daniel Ricciardo
202       Daniil Kvyat
221    Romain Grosjean
223   Pastor Maldonado
```

```
[ ]: len(data[data['quali_pos'] == 0][["date", "GP_name", "position", "quali_pos",␣
      ↪"driver"]])
```

```
[ ]: 57
```

Analysis of difference in qualification position and position

```
[ ]: print(f"Mean: {diff_quali_pos['diff'].mean()}")
     print(f"Standard deviation: {diff_quali_pos['diff'].std()}")
```

```
Mean: -0.18251835930503313
Standard deviation: 2.3588423409618082
```

```
[ ]: print(f"Value counts:\n {diff_quali_pos['diff'].value_counts()}")
     print(f"Value counts in percentage:\n {diff_quali_pos['diff'].
      ↪value_counts(normalize=True)}")
```

```
Value counts:
  0     3831
-1      985
-2      194
 5       89
-3       73
 3       68
 4       58
 2       56
 1       43
-4       28
 10      21
-5       15
-6       12
 6       10
-19       9
-20       9
 7        8
 8        8
 9        8
-10       6
 11       6
-7        5
-14       5
-18       4
-8        3
```

```
-16       3
-12       3
 13       2
 18       2
-17       2
 15       2
-11       2
-24       2
-23       2
-13       2
 16       1
-21       1
-22       1
 23       1
 14       1
 21       1
 17       1
Name: diff, dtype: int64
Value counts in percentage:
  0     0.686190
-1     0.176428
-2     0.034748
 5     0.015941
-3     0.013075
 3     0.012180
 4     0.010389
 2     0.010030
 1     0.007702
-4     0.005015
 10    0.003761
-5     0.002687
-6     0.002149
 6     0.001791
-19    0.001612
-20    0.001612
 7     0.001433
 8     0.001433
 9     0.001433
-10    0.001075
 11    0.001075
-7     0.000896
-14    0.000896
-18    0.000716
-8     0.000537
-16    0.000537
-12    0.000537
 13    0.000358
 18    0.000358
```

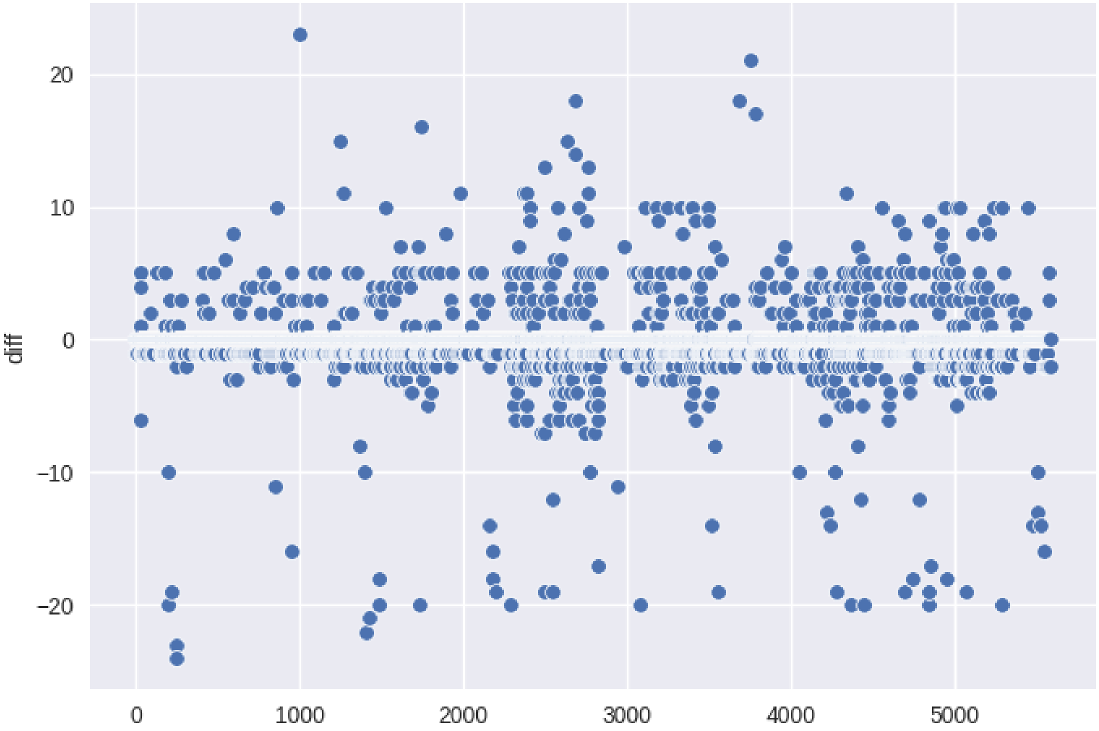
```
-17    0.000358
 15    0.000358
-11    0.000358
-24    0.000358
-23    0.000358
-13    0.000358
 16    0.000179
-21    0.000179
-22    0.000179
 23    0.000179
 14    0.000179
 21    0.000179
 17    0.000179
Name: diff, dtype: float64
```

Box plot of difference of all data

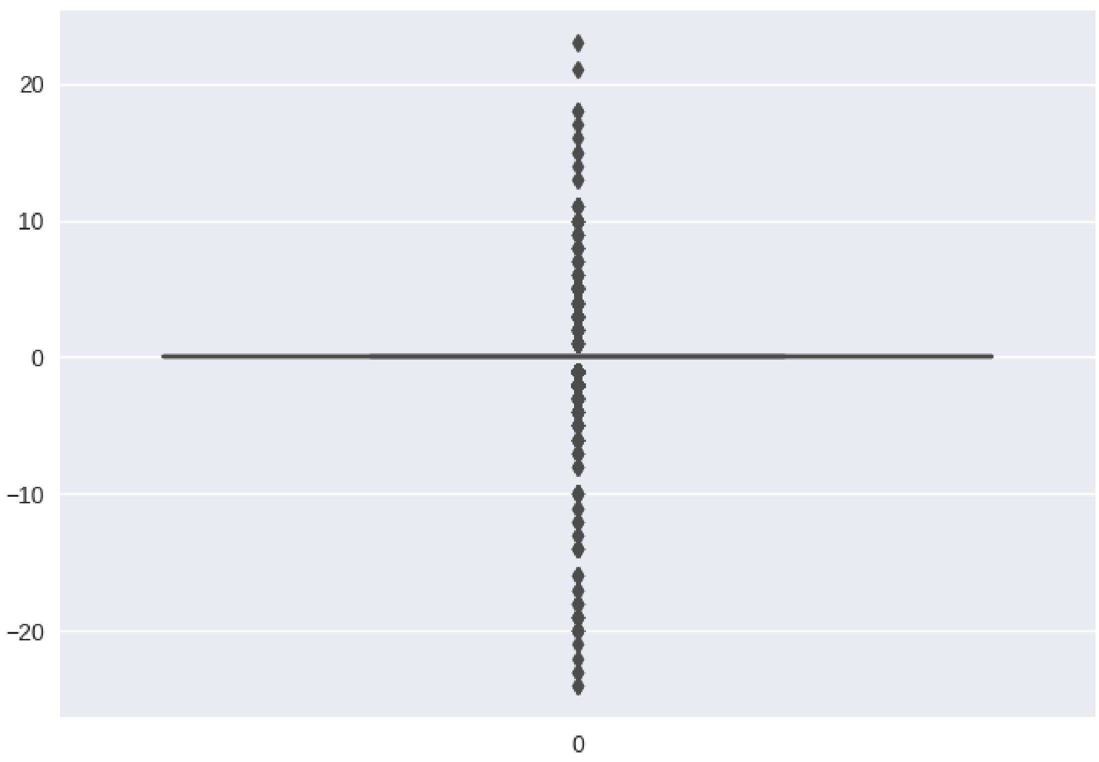
```
[ ]: # Calculate quartiles and IQR
     q1 = diff_quali_pos['diff'].quantile(0.25)
     q3 = diff_quali_pos['diff'].quantile(0.75)
     iqr = q3 - q1

     # Define the threshold for outliers (e.g., 1.5 times the IQR)
     threshold = 1.5

     # Find outliers
     outliers = diff_quali_pos[(diff_quali_pos['diff'] < q1 - threshold * iqr) |␣
      ↪(diff_quali_pos['diff'] > q3 + threshold * iqr)]

     # Print the outliers
     outliers['diff'].unique()
```

```
[ ]: array([ -1,   5,  -6,   1,   4,   2, -20, -10,   3, -19, -23, -24,  -2,
              6,  -3,   8, -11,  10, -16,  23,  15,  11,  -8, -22, -21, -18,
              7,  -4,  16,  -5, -14,   9,  -7,  13, -12,  14,  18, -17,  21,
             17, -13])
```

```
[ ]: sns.scatterplot(diff_quali_pos['diff'])
     plt.show()
```

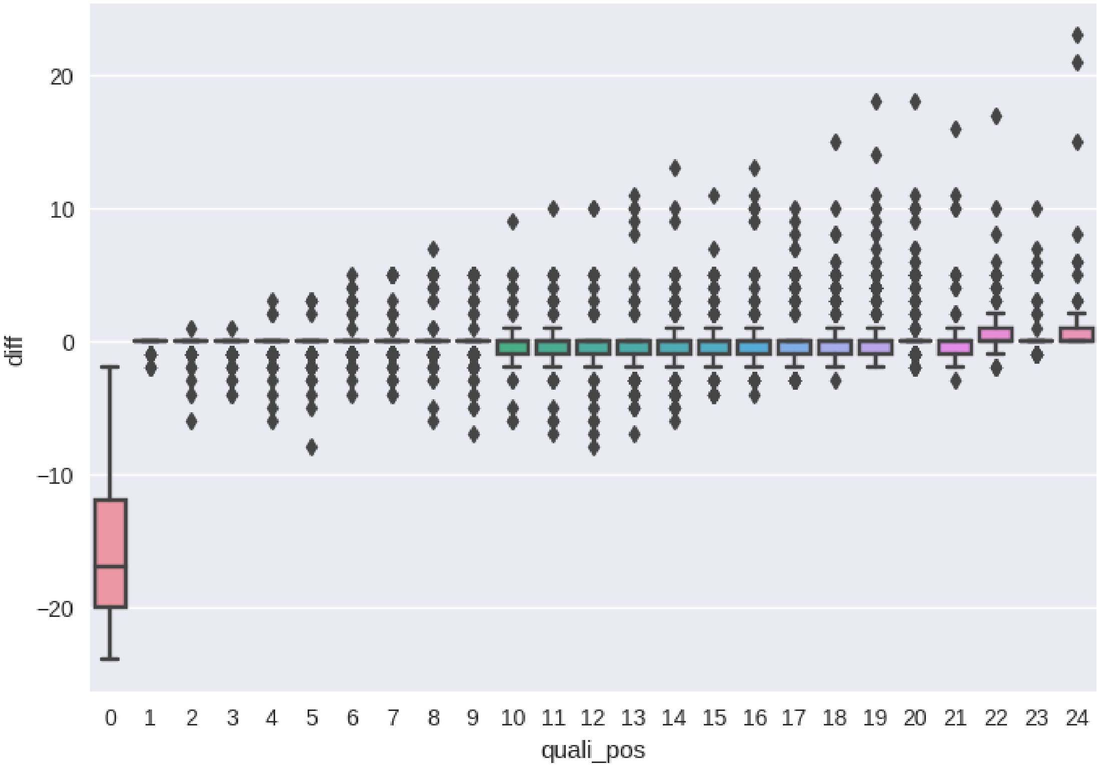


[ ]:
```
sns.boxplot(diff_quali_pos['diff'])
plt.show()
```



Box plot difference by qualification position

```
[ ]: sns.boxplot(diff_quali_pos, x='quali_pos', y='diff')
     plt.show()
```



Analysis of difference of Qualification position and position using only 3 classes

```
[ ]: print(f"Mean of 3 classes: {diff_quali_pos['diff_3_classes'].mean()}")
     print(f"Standard deviation of 3 classes: {diff_quali_pos['diff_3_classes'].
      ↪std()}")
```

```
Mean of 3 classes: -0.019344438473938 74
Standard deviation of 3 classes: 0.29076025596728977
```

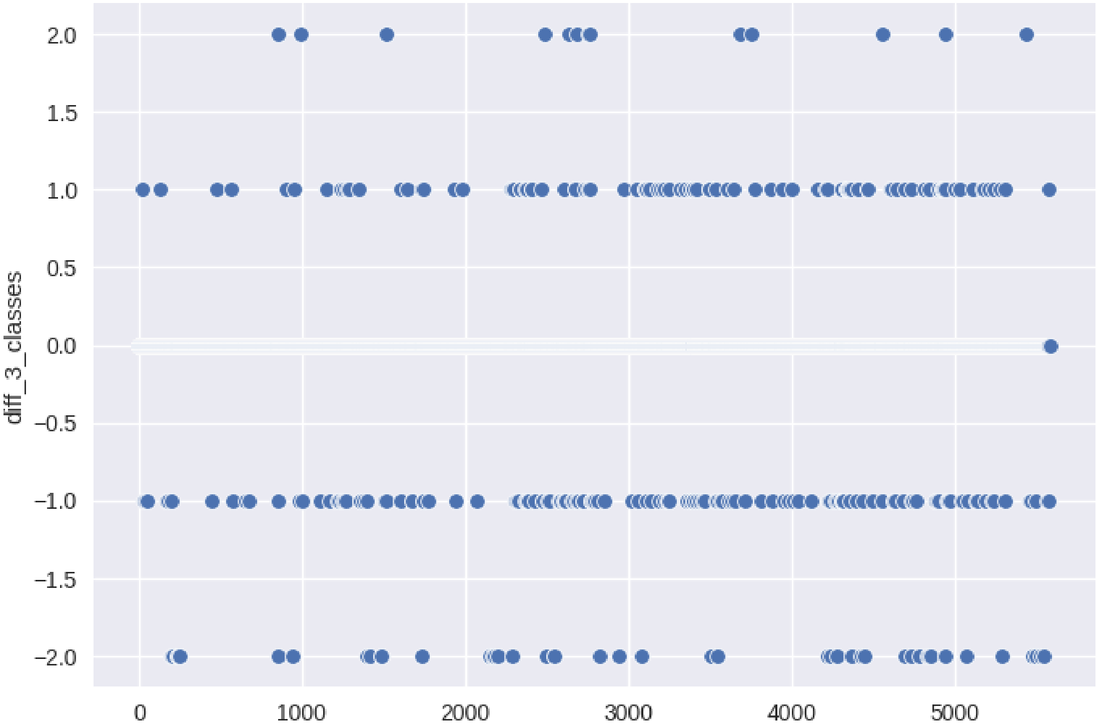
```
[ ]: print(f"Value counts:\n {diff_quali_pos['diff_3_classes'].value_counts()}")
     print(f"Value counts in percentage:\n {diff_quali_pos['diff_3_classes'].
      ↪value_counts(normalize=True)}")
```

```
Value counts:
  0    5283
-1     143
 1      99
-2      45
 2      13
Name: diff_3_classes, dtype: int64
Value counts in percentage:
  0    0.946265
-1    0.025613
```

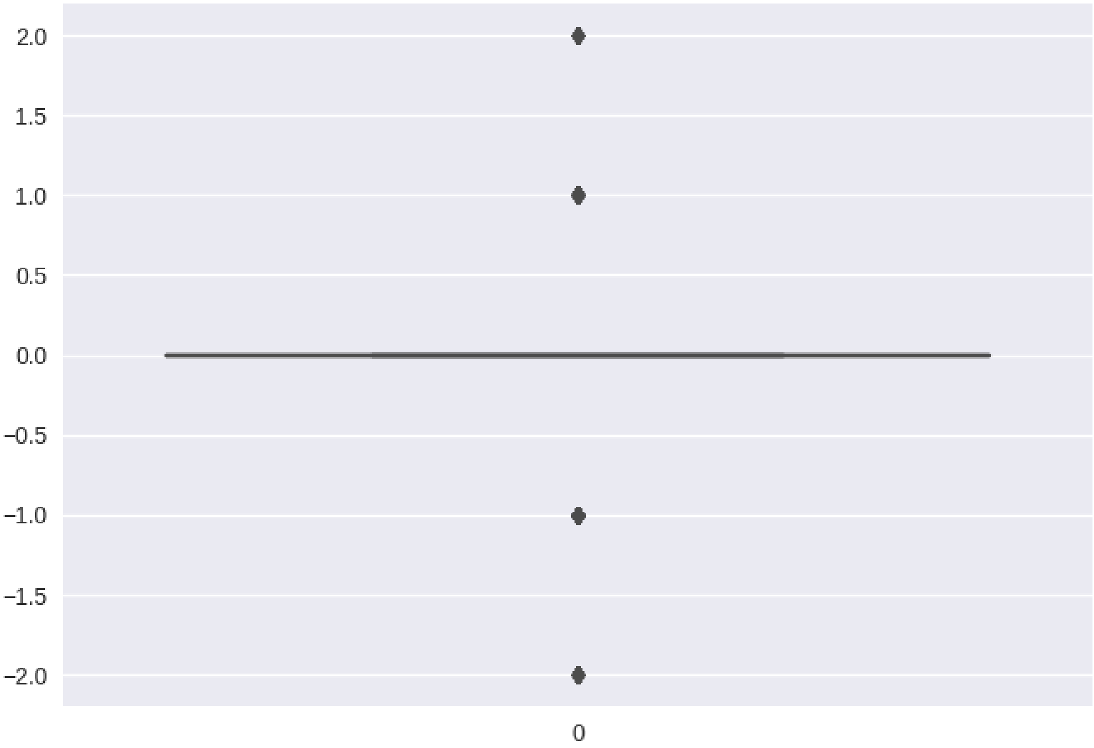
```
 1    0.017732
-2    0.008060
 2    0.002328
Name: diff_3_classes, dtype: float64
```

The models created are really bad, because a good model would get 95% of predictions correct if it only predicts the same position for the race as the qualification position

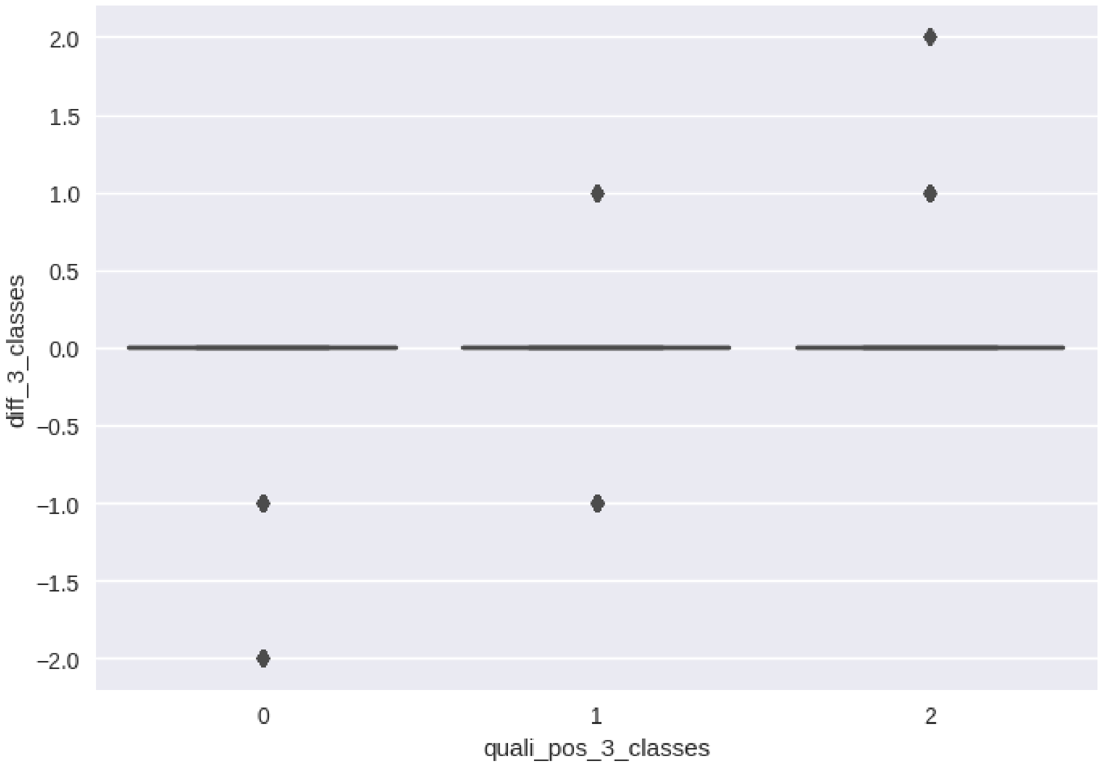
```
[ ]: sns.scatterplot(diff_quali_pos['diff_3_classes'])
     plt.show()
```



```
[ ]: sns.boxplot(diff_quali_pos['diff_3_classes'])
     plt.show()
```



```
[ ]: sns.boxplot(diff_quali_pos, x='quali_pos_3_classes', y='diff_3_classes')
     plt.show()
```



0.4 Show correlation of important variables

- Pearson is linear correlation
- Kendall and spearman are non linear correlations (monotonic relationship)

```
[ ]: data_important =␣
      ↪data[['GP_name','quali_pos','driver','age_at_gp_in_days','position','driver_confidence','ac
      ↪'constructor_reliability']]
     data_important.dtypes
```

```
[ ]: GP_name                        object
     quali_pos                       int64
     driver                         object
     age_at_gp_in_days               int64
     position                        int64
     driver_confidence             float64
     active_driver                   int64
     constructor_reliability       float64
     dtype: object
```

```
[ ]: factorizeColumns(data_important, "GP_name", "FactGP_name")
     factorizeColumns(data_important, "driver", "FactDriver")
```

```
data_important.head()
```

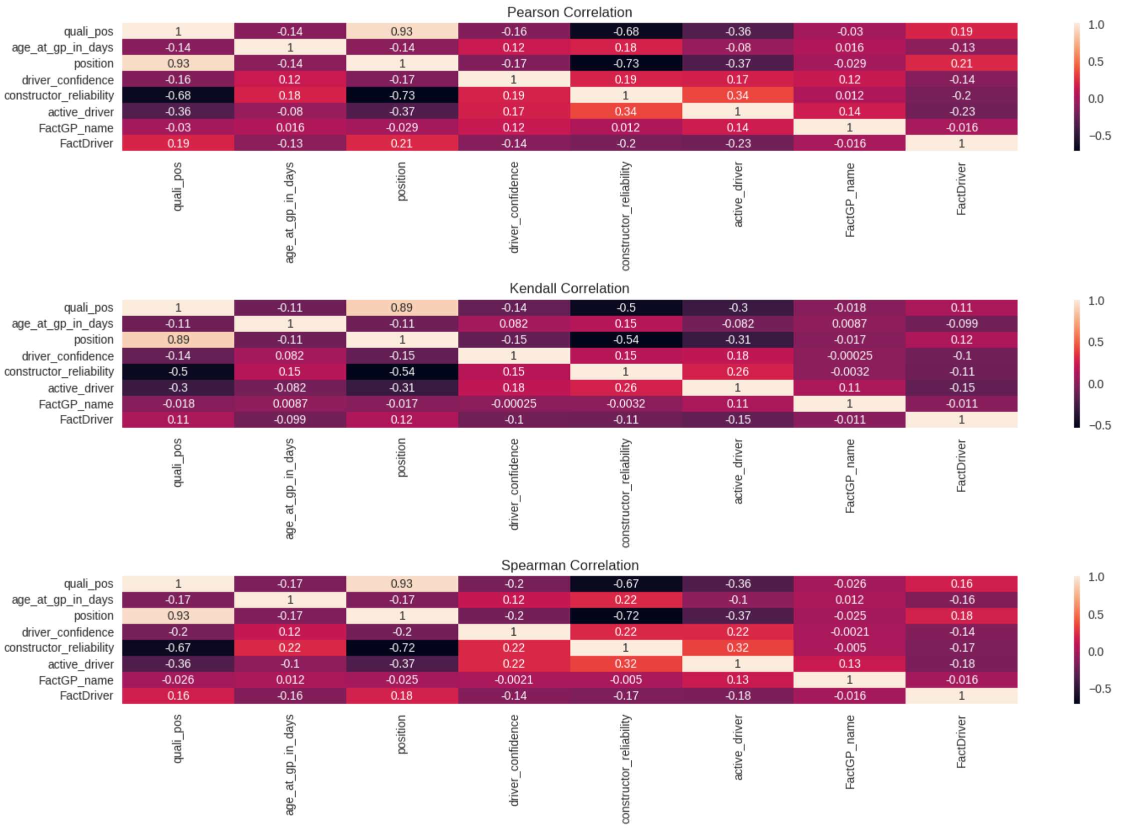
```
[ ]:                          GP_name  quali_pos           driver  \
     0  Albert Park Grand Prix Circuit         18    Nick Heidfeld
     1  Albert Park Grand Prix Circuit          9    Robert Kubica
     2  Albert Park Grand Prix Circuit         11  Nico Hülkenberg
     3  Albert Park Grand Prix Circuit          7  Nico Hülkenberg
     4  Albert Park Grand Prix Circuit         11  Nico Hülkenberg

        age_at_gp_in_days  position  driver_confidence  active_driver  \
     0              12374        18           0.800000              0
     1               9242         9           1.000000              0
     2              10812        12           0.850000              0
     3              11176         8           0.902439              0
     4              11533        11           0.880952              0

        constructor_reliability  FactGP_name  FactDriver
     0                 0.657895            0           0
     1                 0.500000            0           1
     2                 0.467532            0           2
     3                 0.294872            0           2
     4                 0.414634            0           2
```

```
[ ]: new_data_important =␣
      ↪data_important[['quali_pos','age_at_gp_in_days','position','driver_confidence','constructor
     new_data_important.head()
```

```
[ ]:    quali_pos  age_at_gp_in_days  position  driver_confidence  \
     0         18              12374        18           0.800000
     1          9               9242         9           1.000000
     2         11              10812        12           0.850000
     3          7              11176         8           0.902439
     4         11              11533        11           0.880952

        constructor_reliability  active_driver  FactGP_name  FactDriver
     0                 0.657895              0            0           0
     1                 0.500000              0            0           1
     2                 0.467532              0            0           2
     3                 0.294872              0            0           2
     4                 0.414634              0            0           2
```

La correlacion inversa de driver_confidence y constructor reliability nos dice que entre mejor confidence y reliability, mejor posicion tendremos en la carrera

```
[ ]: showCorrelation(new_data_important, nCols=1)
```



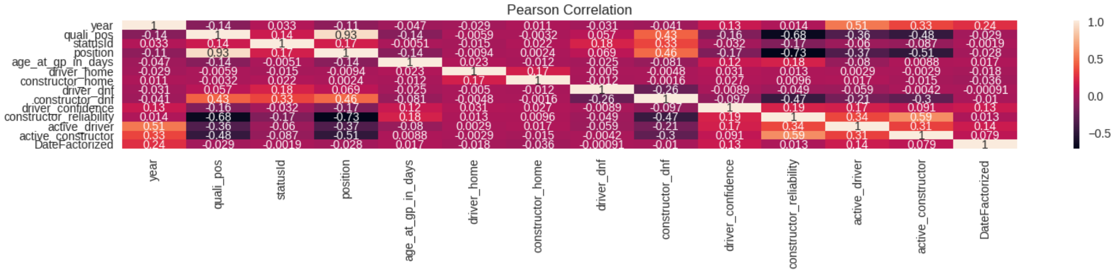
0.5 Do a correlation study of all variables in the dataset

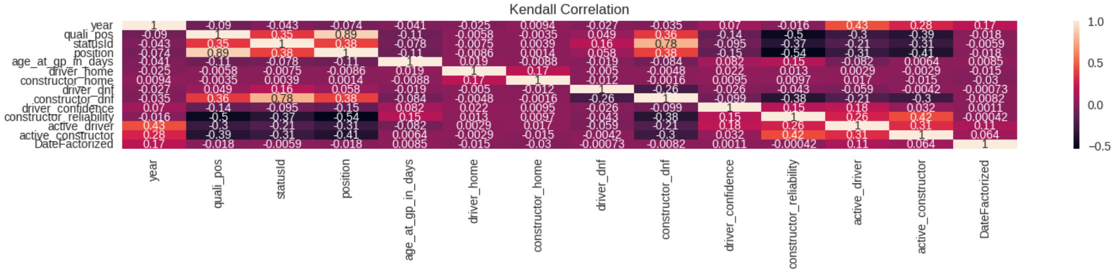
```
[ ]: # data["label"] = pd.factorize(data["driver_nationality"])[0]
     data['DateFactorized'] = pd.factorize(data['date'])[0]
     data["DateFactorized"].head
```

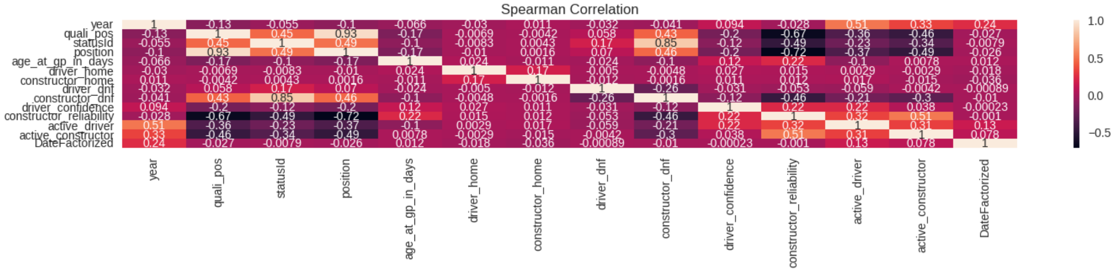
```
[ ]: <bound method NDFrame.head of 0          0
     1          1
     2          2
     3          3
     4          4
              …
     5578     263
     5579     263
     5580     263
     5581     263
     5582     263
     Name: DateFactorized, Length: 5583, dtype: int64>
```

```
[ ]: data.dtypes
[ ]: year                                     int64
     date                            datetime64[ns]
     fp1_date                                object
     fp1_time                                object
     fp2_date                                object
     fp2_time                                object
     fp3_date                                object
     fp3_time                                object
     quali_date                              object
     quali_time                              object
     sprint_date                             object
     sprint_time                             object
     quali_pos                                int64
     statusId                                 int64
     position                                 int64
     dob                                     object
     driver_nationality                      object
     constructor                             object
     constructor_nationality                 object
     GP_name                                 object
     country                                 object
     driver                                  object
     age_at_gp_in_days                        int64
     driver_home                              int64
     constructor_home                         int64
     driver_dnf                               int64
     constructor_dnf                          int64
     driver_confidence                      float64
     constructor_reliability                float64
     active_driver                            int64
     active_constructor                       int64
     DateFactorized                           int64
     dtype: object
```

```
[ ]: showCorrelation(data, nCols=1)
```


Pearson Correlation
year 1 -0.14 0.033 -0.11 -0.047 -0.029 0.011 -0.031 -0.041 0.13 0.014 0.51 0.33 0.24
quali_pos -0.14 1 0.14 0.93 -0.14 -0.0059 -0.0032 0.057 0.43 -0.16 -0.68 -0.36 -0.48 -0.029
statusId 0.033 0.14 1 0.17 -0.0051 -0.015 0.022 0.18 0.33 -0.032 -0.17 -0.06 -0.087 -0.0019
position -0.11 0.93 0.17 1 -0.14 -0.0094 0.0024 0.069 0.46 -0.17 -0.73 -0.37 -0.51 -0.028
age_at_gp_in_days -0.047 -0.14 -0.0051 -0.14 1 0.023 -0.012 -0.025 -0.081 0.12 0.18 -0.08 0.0088 0.017
driver_home -0.029 -0.0059 -0.015 -0.0094 0.023 1 0.17 -0.005 -0.0048 0.031 0.013 0.0029 -0.0029 -0.018
constructor_home 0.011 -0.0032 0.022 0.0024 -0.012 0.17 1 -0.012 -0.0016 0.027 0.0096 0.017 -0.015 -0.036
driver_dnf -0.031 0.057 0.18 0.069 -0.025 -0.005 -0.012 1 -0.26 -0.0089 -0.049 -0.059 -0.0042 -0.00091
constructor_dnf -0.041 0.43 0.33 0.46 -0.081 -0.0048 -0.0016 -0.26 1 -0.097 -0.47 -0.21 -0.3 -0.01
driver_confidence 0.13 -0.16 -0.032 -0.17 0.12 0.031 0.027 -0.0089 -0.097 1 0.19 0.17 0.091 0.13
constructor_reliability 0.014 -0.68 -0.17 -0.73 0.18 0.013 0.0096 -0.049 -0.47 0.19 1 0.34 0.59 0.013
active_driver 0.51 -0.36 -0.06 -0.37 -0.08 0.0029 0.017 -0.059 -0.21 0.17 0.34 1 0.31 0.14
active_constructor 0.33 -0.48 -0.087 -0.51 0.0088 -0.0029 -0.015 -0.0042 -0.3 0.091 0.59 0.31 1 0.079
DateFactorized 0.24 -0.029 -0.0019 -0.028 0.017 -0.018 -0.036 -0.00091 -0.01 0.13 0.013 0.14 0.079 1
year
quali_pos
statusId
position
age_at_gp_in_days
driver_home
constructor_home
driver_dnf
constructor_dnf
driver_confidence
constructor_reliability
active_driver
active_constructor
DateFactorized
1.0
0.5
0.0
−0.5


Kendall Correlation
year 1 -0.09 -0.043 -0.074 -0.041 -0.025 0.0094 -0.027 -0.035 0.07 -0.016 0.43 0.28 0.17
quali_pos -0.09 1 0.35 0.89 -0.11 -0.0058 -0.0035 0.049 0.36 -0.14 -0.5 -0.3 -0.39 -0.018
statusId -0.043 0.35 1 0.38 -0.078 -0.0075 0.0039 0.16 0.78 -0.095 -0.37 -0.21 -0.31 -0.0059
position -0.074 0.89 0.38 1 -0.11 -0.0086 0.0014 0.058 0.38 -0.15 -0.54 -0.31 -0.41 -0.018
age_at_gp_in_days -0.041 -0.11 -0.078 -0.11 1 0.019 -0.0088 -0.019 -0.084 0.082 0.15 -0.082 0.0064 0.0085
driver_home -0.025 -0.0058 -0.0075 -0.0086 0.019 1 0.17 -0.005 -0.0048 0.022 0.013 0.0029 -0.0029 -0.015
constructor_home 0.0094 -0.0035 0.0039 0.0014 -0.0088 0.17 1 -0.012 -0.0016 0.0095 0.0097 0.017 -0.015 -0.03
driver_dnf -0.027 0.049 0.16 0.058 -0.019 -0.005 -0.012 1 -0.26 -0.026 -0.043 -0.059 -0.0042 -0.00073
constructor_dnf -0.035 0.36 0.78 0.38 -0.084 -0.0048 -0.0016 -0.26 1 -0.099 -0.38 -0.21 -0.3 -0.0082
driver_confidence 0.07 -0.14 -0.095 -0.15 0.082 0.022 0.0095 -0.026 -0.099 1 0.15 0.18 0.032 0.0011
constructor_reliability -0.016 -0.5 -0.37 -0.54 0.15 0.013 0.0097 -0.043 -0.38 0.15 1 0.26 0.42 -0.00042
active_driver 0.43 -0.3 -0.21 -0.31 -0.082 0.0029 0.017 -0.059 -0.21 0.18 0.26 1 0.31 0.11
active_constructor 0.28 -0.39 -0.31 -0.41 0.0064 -0.0029 -0.015 -0.0042 -0.3 0.032 0.42 0.31 1 0.064
DateFactorized 0.17 -0.018 -0.0059 -0.018 0.0085 -0.015 -0.03 -0.00073 -0.0082 0.0011 -0.00042 0.11 0.064 1
year
quali_pos
statusId
position
age_at_gp_in_days
driver_home
constructor_home
driver_dnf
constructor_dnf
driver_confidence
constructor_reliability
active_driver
active_constructor
DateFactorized
1.0
0.5
0.0
−0.5


Spearman Correlation
year 1 -0.13 -0.055 -0.1 -0.066 -0.03 0.011 -0.032 -0.041 0.094 -0.028 0.51 0.33 0.24
quali_pos -0.13 1 0.45 0.93 -0.17 -0.0069 -0.0042 0.058 0.43 -0.2 -0.67 -0.36 -0.46 -0.027
statusId -0.055 0.45 1 0.49 -0.1 -0.0083 0.0043 0.17 0.85 -0.12 -0.49 -0.23 -0.34 -0.0079
position -0.1 0.93 0.49 1 -0.17 -0.01 0.0016 0.07 0.46 -0.2 -0.72 -0.37 -0.49 -0.026
age_at_gp_in_days -0.066 -0.17 -0.1 -0.17 1 0.024 -0.011 -0.024 -0.1 0.12 0.22 -0.1 0.0078 0.012
driver_home -0.03 -0.0069 -0.0083 -0.01 0.024 1 0.17 -0.005 -0.0048 0.027 0.015 0.0029 -0.0029 -0.018
constructor_home 0.011 -0.0042 0.0043 0.0016 -0.011 0.17 1 -0.012 -0.0016 0.011 0.012 0.017 -0.015 -0.036
driver_dnf -0.032 0.058 0.17 0.07 -0.024 -0.005 -0.012 1 -0.26 -0.031 -0.053 -0.059 -0.0042 -0.00089
constructor_dnf -0.041 0.43 0.85 0.46 -0.1 -0.0048 -0.0016 -0.26 1 -0.12 -0.46 -0.21 -0.3 -0.01
driver_confidence 0.094 -0.2 -0.12 -0.2 0.12 0.027 0.011 -0.031 -0.12 1 0.22 0.22 0.038 -0.00023
constructor_reliability -0.028 -0.67 -0.49 -0.72 0.22 0.015 0.012 -0.053 -0.46 0.22 1 0.32 0.51 -0.001
active_driver 0.51 -0.36 -0.23 -0.37 -0.1 0.0029 0.017 -0.059 -0.21 0.22 0.32 1 0.31 0.13
active_constructor 0.33 -0.46 -0.34 -0.49 0.0078 -0.0029 -0.015 -0.0042 -0.3 0.038 0.51 0.31 1 0.078
DateFactorized 0.24 -0.027 -0.0079 -0.026 0.012 -0.018 -0.036 -0.00089 -0.01 -0.00023 -0.001 0.13 0.078 1
year
quali_pos
statusId
position
age_at_gp_in_days
driver_home
constructor_home
driver_dnf
constructor_dnf
driver_confidence
constructor_reliability
active_driver
active_constructor
DateFactorized
1.0
0.5
0.0
−0.5