STØTTET AF
Promilleafgiftsfonden for landbrug

Report

November 27, 2023

Created by: Jakob Astrup Vestergaard & Lasse Rose Malskær

This notebook details an exploratory, and statistical analysis for Carbon in the ground.

1 Setup

1.0.1 Data

The following data file have been used for the analysis of this project.

- pm_22_8512_ap1_Datasaet.xlsx

which consists of two data sources:

- Mark Analysis Online (MAO)
- Forsøg DB (FSDB)

For the exploratory and statistical analysis we have used the data from MAO, which contains 36774 data observations and 122 columns

the columns of interest are:

- 'totalkulstofpct' contains the percentage of carbon for each observation
- 'totalnpct' contains the percentage of nitrogen for each observation
- 'jb_ny' contains the soil sample for each observation (values from 1-11)
- 'ler' contains the amount of clay for each observation
- 'afgroede_2011' - 'afgroede_2022' each of these columns contains the crop grown in the field for that year

1.0.2 Exploratory data analysis

Findings from exploratory data analysis Through the exploratory data analysis it was found that the column containing 'totalkulstofpct' had multiple observations missing data. the same issue was identified for both 'totalnpct' and 'ler'.

Besides the missing data, it was found that 'jb_ny' contained a lot of 0-values, which is misleading because we only categorize soil samples from 1-11.

Solutions to some of the problems found in the exploratory data analysis For the problem regarding 'totalkulstofpct' which is our variable of interest, we found that through the values in the column 'humus' we could calculate the missing values in 'totalkulstofpct' with a factor 0.5813953488372093. This was found through consultation with Julie from "Planter & Miljø".

We had to pull new soil samples to account for the 0-values found in the 'jb_ny' column, the original data were replaced with these new values, which ensured that there are no 0-values in this column.

1.1 Statistical Analysis

1.1.1 Main tasks

We had the task of categorising each observation based on the crop rotation up until the year of the sample. The categories included: - Category 0: Unknown - Category 1: Clover in the crop rotation - Category 2: Corn with "rotation crops" - Category 3: Grains, continous - Category 4: Grains with "rotation crops"

We found that the number of observations for each category was as follows:

- Category 0: 9137
- Category 1: 19864
- Category 2: 4728
- Category 3: 1108
- Category 4: 1916

Some of the main tasks also consists of checking the conditions for regression, which includes linearity and correlation.

We concluded the statistical analysis for the whole dataset by performing ols regression, but because of ambiguous results we decided to create subsets of the dataset based on soil sample numbers, while also create cleaned versions for each of these datasets based on 0-values in both nitrogen and clay. We then performed the same checks of conditions for each subset of the dataset. Lastly we created means for each of these subdatasets, as well as mean and standard deviation for each crop rotation

2 Technical setup

2.0.1 Conda Environment

In this notebook the py39_pla_sathub conda environment is used. it can be installed using the environment.yml file

Imports

[ ]:
```
import pandas as pd
import geopandas as gpd
import matplotlib.pyplot as plt
import numpy as np
import hvplot.pandas  # noqa
import hvplot.xarray  # noqa
from shapely.wkt import loads
from shapely import wkt
import statsmodels.api as sm
from statsmodels.formula.api import ols
import seaborn as sns
from scipy import stats
```

```
from tabulate import tabulate
```

2.0.2 Data import and initial data handling

[ ]:
```
df = pd.read_excel('data/pm_22_8512_ap1_Datasaet.xlsx') # Local data folder

mao_df = df[df['dataset'] == 'MAO']

mao_df['geometry'] = mao_df['geometry'].apply(loads)

gdf = gpd.GeoDataFrame(mao_df, geometry = 'geometry')

gdf = gdf.rename(columns={'index_right': 'index_right_renamed'})

gdf = gdf.set_crs('epsg:4326')
```

```
/tmp/ipykernel_738/3436566046.py:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  mao_df['geometry'] = mao_df['geometry'].apply(loads)
```

Pulling new soil sample numbers

[ ]:
```
jb = gpd.read_file('data/jb/Jordbundskort_2019.TAB')

jb = jb.to_crs(4326)
```

Performing a spacial join to initialize new soil sample numbers

[ ]:
```
joined = gpd.sjoin(gdf, jb, how = "left", op = "within")
matching_indices = joined.loc[~joined["index_right"].isna(), "index_right"].
 ↪astype(int)

for i in matching_indices.index:
    gdf.loc[i, 'jb_ny'] = jb.loc[matching_indices[i], 'JB_nr']
```

```
/home/jaav/miniconda3/envs/py39_pla_sathub/lib/python3.9/site-
packages/IPython/core/interactiveshell.py:3466: FutureWarning: The `op`
parameter is deprecated and will be removed in a future release. Please use the
`predicate` parameter instead.
  if await self.run_code(code, result, async_=asy):
```

[ ]:
```
for i in gdf.index:
    if gdf.loc[i, 'totalkulstofpct'] == 0:
        gdf.loc[i, 'totalkulstofpct'] = gdf.loc[i, 'humus'] * 0.5813953488372093
```

Categorising based on crop rotation

[ ]:
```
afgroede_year = {
    2011: 'afgroede_2011',
    2012: 'afgroede_2012',
    2013: 'afgroede_2013',
    2014: 'afgroede_2014',
    2015: 'afgroede_2015',
    2016: 'afgroede_2016',
    2017: 'afgroede_2017',
    2018: 'afgroede_2018',
    2019: 'afgroede_2019',
    2020: 'afgroede_2020',
    2021: 'afgroede_2021',
    2022: 'afgroede_2022'
}

kloever = [260, 268, 264, 276, 267, 266, 255,
           261, 265, 174, 256, 173, 285]

majs = [216, 5, 423, 19]

vekselafgoeder = [30, 22, 424, 215, 21]

korn = [230, 701, 702, 703, 704, 705, 234, 235,
        706, 707, 708, 709, 710, 220, 221, 222,
        223, 224, 210, 211, 212, 213, 214, 217,
        1, 2, 6, 3, 55, 7, 18, 10, 11, 13, 57,
        14, 15, 9]

for index, row in gdf.iterrows():
    for year, variable_name in afgroede_year.items():
        if row['aar_udtagning'] == year:
            afgroede = list()
            for i in range(2011, year + 1):
                column_name = 'afgroede_' + str(i)
                afgroede.append(gdf.loc[index, column_name])

            matching_count_majs = sum(value in majs for value in afgroede)
            cutoff_majs = round(0.8*len(afgroede))
            missing_values_korn = [value for value in afgroede if value not in␣
 ↪korn]
            matching_count_korn = sum(value in korn for value in afgroede)
            cutoff_korn = round(0.7*len(afgroede))

            if any(value in kloever for value in afgroede):
                gdf.loc[index, 'saedskifte'] = 1
            elif matching_count_majs >= cutoff_majs:
```

```
                gdf.loc[index, 'saedskifte'] = 2
            elif all(value in korn for value in afgroede):
                gdf.loc[index, 'saedskifte'] = 3
            elif matching_count_korn >= cutoff_korn and all(value in␣
↪vekselafgoeder for value in missing_values_korn):
                gdf.loc[index, 'saedskifte'] = 4
            else:
                gdf.loc[index, 'saedskifte'] = 0
```

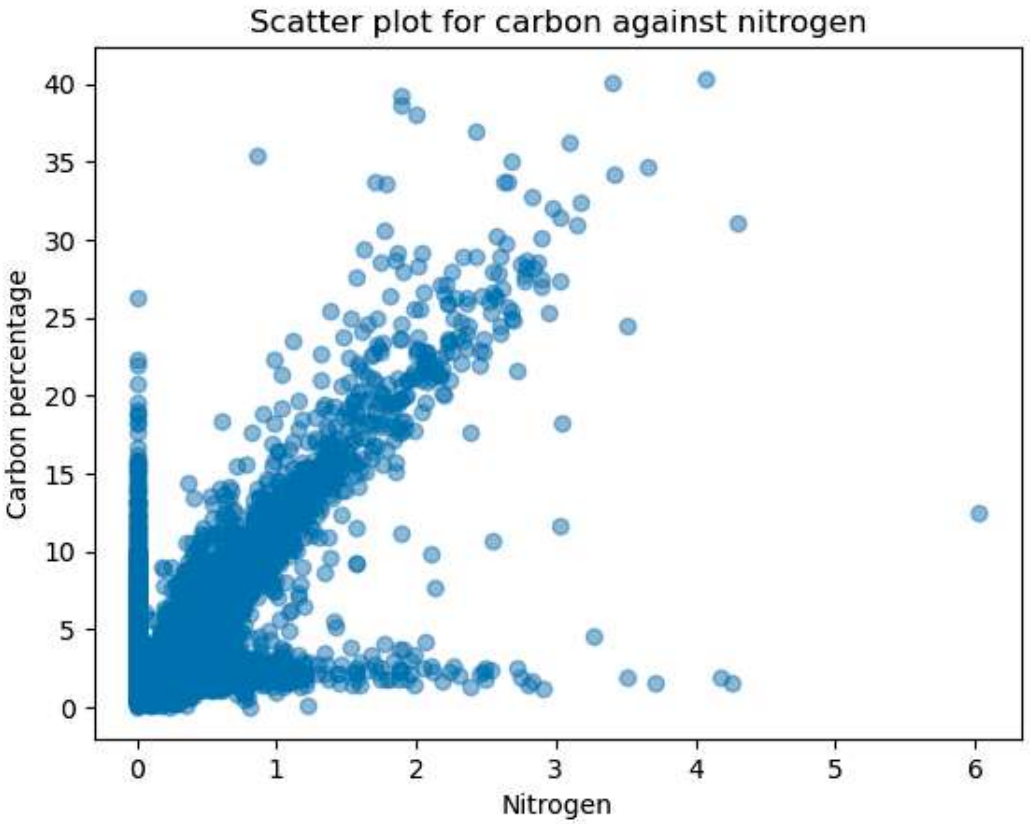
Creating cleaned sub-datasets based on 0-values in both nitrogen and clay

[ ]:
```
clay = gdf[gdf['ler'] != 0]

nitrogen = gdf[gdf['totalnpct'] != 0]
```

Linearity and Correlation is checked for the original dataset.

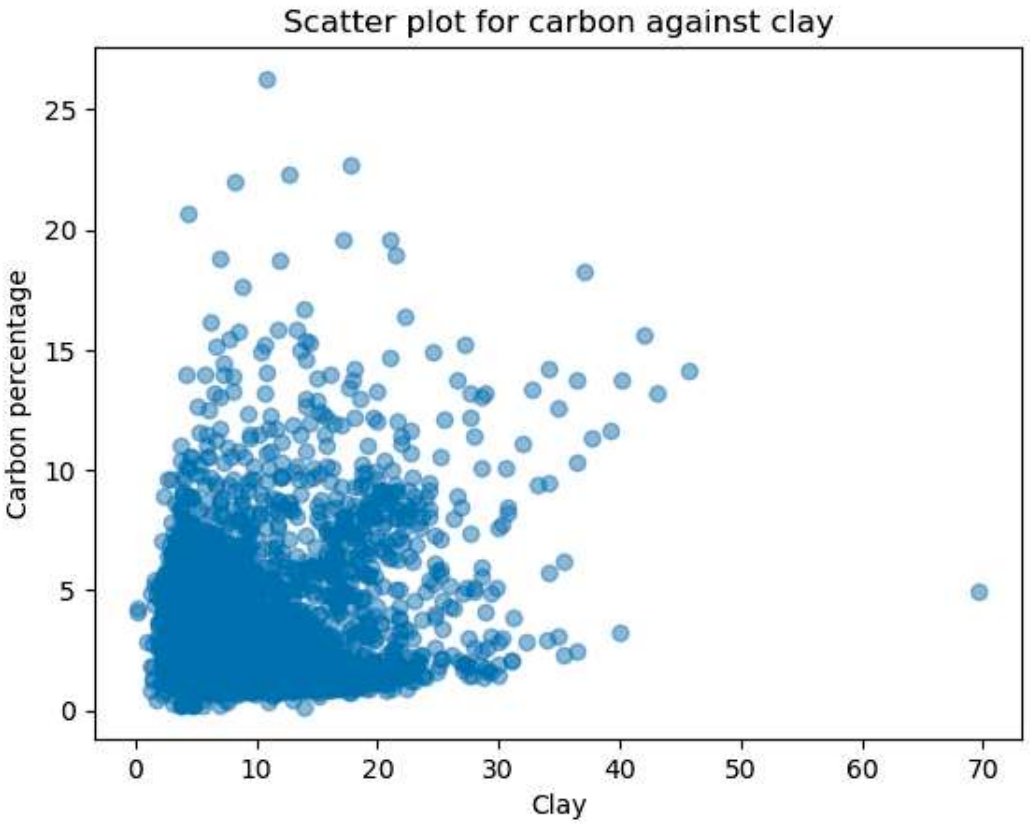
[ ]:
```
plt.scatter(gdf['totalnpct'], gdf['totalkulstofpct'], alpha = 0.5)
plt.title('Scatter plot for carbon against nitrogen')
plt.xlabel('Nitrogen')
plt.ylabel('Carbon percentage')
plt.show
```

[ ]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



[ ]:
```
plt.scatter(clay['ler'], clay['totalkulstofpct'], alpha = 0.5)
plt.title('Scatter plot for carbon against clay')
plt.xlabel('Clay')
plt.ylabel('Carbon percentage')
plt.show
```

[ ]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



[ ]:
```
plt.figure(figsize = (10,5))
sns.heatmap(gdf[['totalkulstofpct', 'ler', 'totalnpct']].corr(), vmin = -1,␣
↪annot = True)
```

[ ]: <Axes: >

We find some linearity between Carbon and Nitrogen, while Carbon and Clay reveals poor linearity, likewise for correlation.

Regression analysis on the original dataset For the regression we are using the variables 'totalnpct', 'ler', 'jb_ny' and 'saedskifte' as regressors to explain evident fluctuations in 'totalkulstofpct'.

[ ]:
```
model = ols(formula= 'totalkulstofpct ~ totalnpct + ler + C(jb_ny) +␣
↪C(saedskifte)', data = gdf)

result = model.fit()

print(result.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:        totalkulstofpct   R-squared:                       0.649
Model:                            OLS   Adj. R-squared:                  0.649
Method:                 Least Squares   F-statistic:                     4857.
Date:                Mon, 27 Nov 2023   Prob (F-statistic):               0.00
Time:                        08:22:35   Log-Likelihood:                -66310.
No. Observations:               36752   AIC:                         1.327e+05
Df Residuals:                   36737   BIC:                         1.328e+05
Df Model:                          14
Covariance Type:            nonrobust
==============================================================================
======
```

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1.1940	0.022	54.559	0.000	1.151	1.237
C(jb_ny)[T.2.0]	-0.3254	0.022	-14.767	0.000	-0.369	-0.282
C(jb_ny)[T.3.0]	-0.2409	0.037	-6.591	0.000	-0.313	-0.169
C(jb_ny)[T.4.0]	-0.2561	0.023	-11.261	0.000	-0.301	-0.212
C(jb_ny)[T.5.0]	-0.4106	0.085	-4.849	0.000	-0.577	-0.245
C(jb_ny)[T.6.0]	-0.5939	0.028	-21.017	0.000	-0.649	-0.538
C(jb_ny)[T.7.0]	-0.6352	0.078	-8.169	0.000	-0.788	-0.483
C(jb_ny)[T.8.0]	0.5377	0.115	4.674	0.000	0.312	0.763
C(jb_ny)[T.11.0]	2.4922	0.033	75.771	0.000	2.428	2.557
C(saedskifte)[T.1]	0.0660	0.019	3.434	0.001	0.028	0.104
C(saedskifte)[T.2]	-0.2241	0.027	-8.299	0.000	-0.277	-0.171
C(saedskifte)[T.3]	0.2433	0.047	5.160	0.000	0.151	0.336
C(saedskifte)[T.4]	-0.4316	0.041	-10.435	0.000	-0.513	-0.351
totalnpct	6.6501	0.035	188.389	0.000	6.581	6.719
ler	0.0809	0.002	33.483	0.000	0.076	0.086

Omnibus:	14342.540	Durbin-Watson:	1.975
Prob(Omnibus):	0.000	Jarque-Bera (JB):	3656382.487
Skew:	0.664	Prob(JB):	0.00
Kurtosis:	51.846	Cond. No.	65.6

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

2.0.3 Subsets of the dataset based on soil sample numbers

These subsets has also been created for the cleaned dataframes based on 0-values in nitrogen and clay.

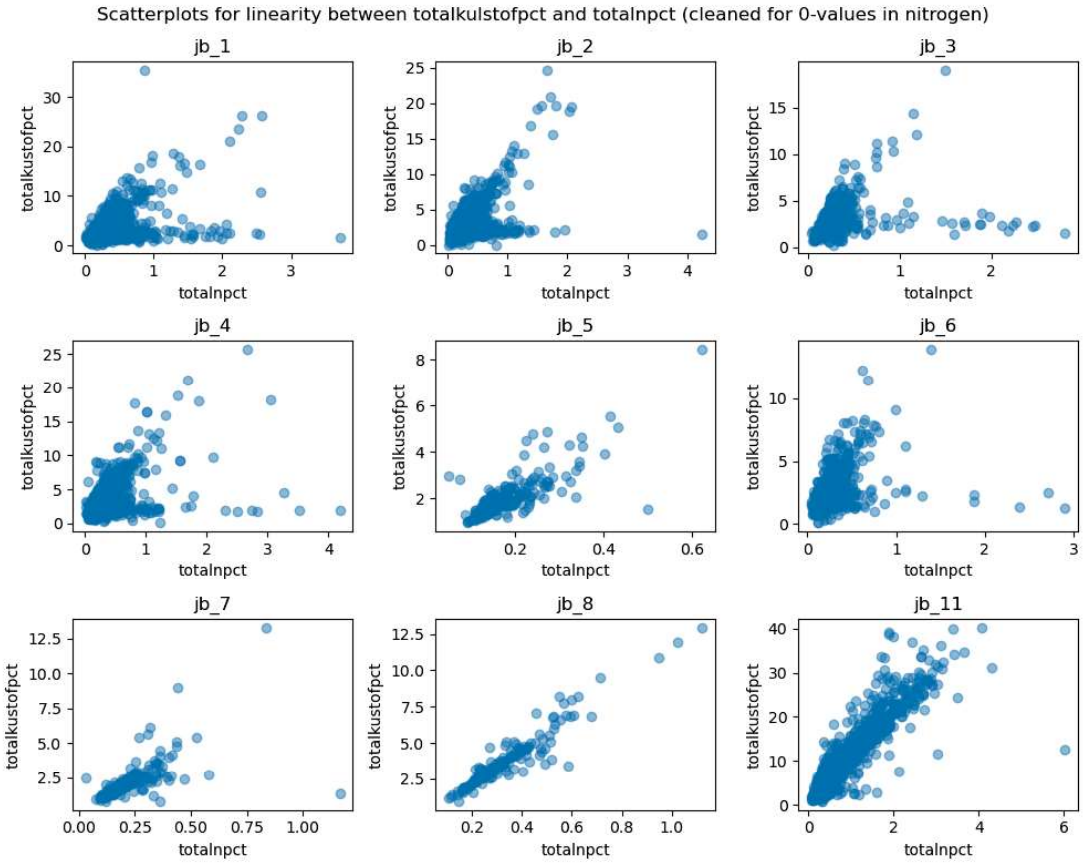
[ ]:
```
jb_dataframes = {
    'jb_1_df': gdf[gdf['jb_ny'] == 1],
    'jb_2_df': gdf[gdf['jb_ny'] == 2],
    'jb_3_df': gdf[gdf['jb_ny'] == 3],
    'jb_4_df': gdf[gdf['jb_ny'] == 4],
    'jb_5_df': gdf[gdf['jb_ny'] == 5],
    'jb_6_df': gdf[gdf['jb_ny'] == 6],
    'jb_7_df': gdf[gdf['jb_ny'] == 7],
    'jb_8_df': gdf[gdf['jb_ny'] == 8],
    'jb_11_df': gdf[gdf['jb_ny'] == 11]
}

nitrogen_dataframes = {
    'jb_1_df_nitrogen': nitrogen[nitrogen['jb_ny'] == 1],
    'jb_2_df_nitrogen': nitrogen[nitrogen['jb_ny'] == 2],
    'jb_3_df_nitrogen': nitrogen[nitrogen['jb_ny'] == 3],
    'jb_4_df_nitrogen': nitrogen[nitrogen['jb_ny'] == 4],
    'jb_5_df_nitrogen': nitrogen[nitrogen['jb_ny'] == 5],
    'jb_6_df_nitrogen': nitrogen[nitrogen['jb_ny'] == 6],
    'jb_7_df_nitrogen': nitrogen[nitrogen['jb_ny'] == 7],
    'jb_8_df_nitrogen': nitrogen[nitrogen['jb_ny'] == 8],
    'jb_11_df_nitrogen': nitrogen[nitrogen['jb_ny'] == 11]
}

clay_dataframes = {
    'jb_1_df_clay': clay[clay['jb_ny'] == 1],
    'jb_2_df_clay': clay[clay['jb_ny'] == 2],
    'jb_3_df_clay': clay[clay['jb_ny'] == 3],
    'jb_4_df_clay': clay[clay['jb_ny'] == 4],
    'jb_5_df_clay': clay[clay['jb_ny'] == 5],
    'jb_6_df_clay': clay[clay['jb_ny'] == 6],
    'jb_7_df_clay': clay[clay['jb_ny'] == 7],
    'jb_8_df_clay': clay[clay['jb_ny'] == 8],
    'jb_11_df_clay': clay[clay['jb_ny'] == 11]
}
```

Condition checks for each of the subsets

Linearity

[ ]:
```
fig, axs = plt.subplots(3, 3, figsize = (10,8))
```

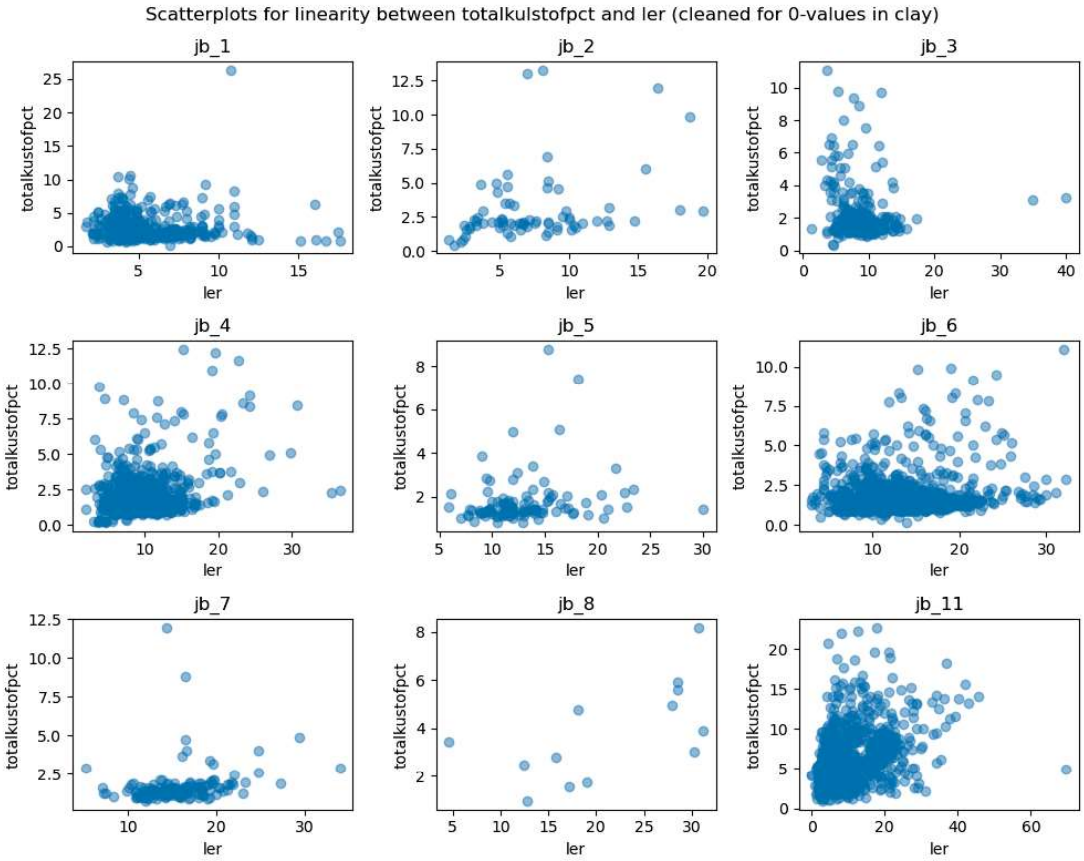
```
fig.suptitle('Scatterplots for linearity between totalkulstofpct and totalnpct␣
 ↪(cleaned for 0-values in nitrogen)')

for i in range(3):
    for j in range(3):
        idx = i * 3 + j + 1
        if idx <= 8:
            dataframe_name = f'jb_{idx}_df_nitrogen'
            title = f'jb_{idx}'
        else:
            dataframe_name = 'jb_11_df_nitrogen'
            title = 'jb_11'

        axs[i,j].scatter(nitrogen_dataframes[dataframe_name]['totalnpct'],␣
 ↪nitrogen_dataframes[dataframe_name]['totalkulstofpct'], alpha = 0.5)
        axs[i,j].set_title(title)
        axs[i,j].set_ylabel('totalkustofpct')
        axs[i,j].set_xlabel('totalnpct')

plt.tight_layout()

plt.show()
```



The data shows quite well linearity for the soil sample numbers 5,7,8 and 11, while still showing some kind of linearity for the remainder soil sample numbers.

[ ]:
```
fig, axs = plt.subplots(3, 3, figsize = (10,8))

fig.suptitle('Scatterplots for linearity between totalkulstofpct and ler␣
 ↪(cleaned for 0-values in clay)')

for i in range(3):
    for j in range(3):
        idx = i * 3 + j + 1
        if idx <= 8:
            dataframe_name = f'jb_{idx}_df_clay'
            title = f'jb_{idx}'
        else:
            dataframe_name = 'jb_11_df_clay'
            title = 'jb_11'
```

```
        axs[i,j].scatter(clay_dataframes[dataframe_name]['ler'],␣
↪clay_dataframes[dataframe_name]['totalkulstofpct'], alpha = 0.5)
        axs[i,j].set_title(title)
        axs[i,j].set_xlabel('ler')
        axs[i,j].set_ylabel('totalkustofpct')

plt.tight_layout()

plt.show()
```

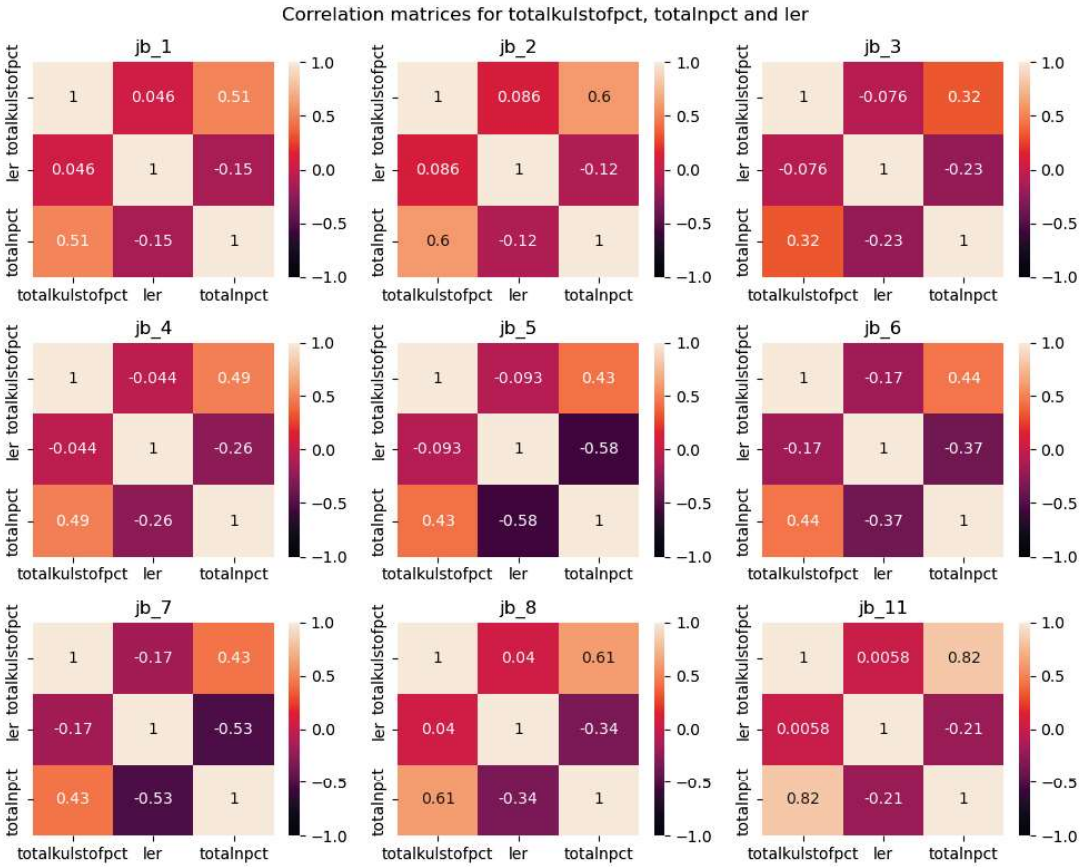


The data reveals poor linearity between 'totalkustofpct' and 'ler' over all soil sample numbers.

Correlation

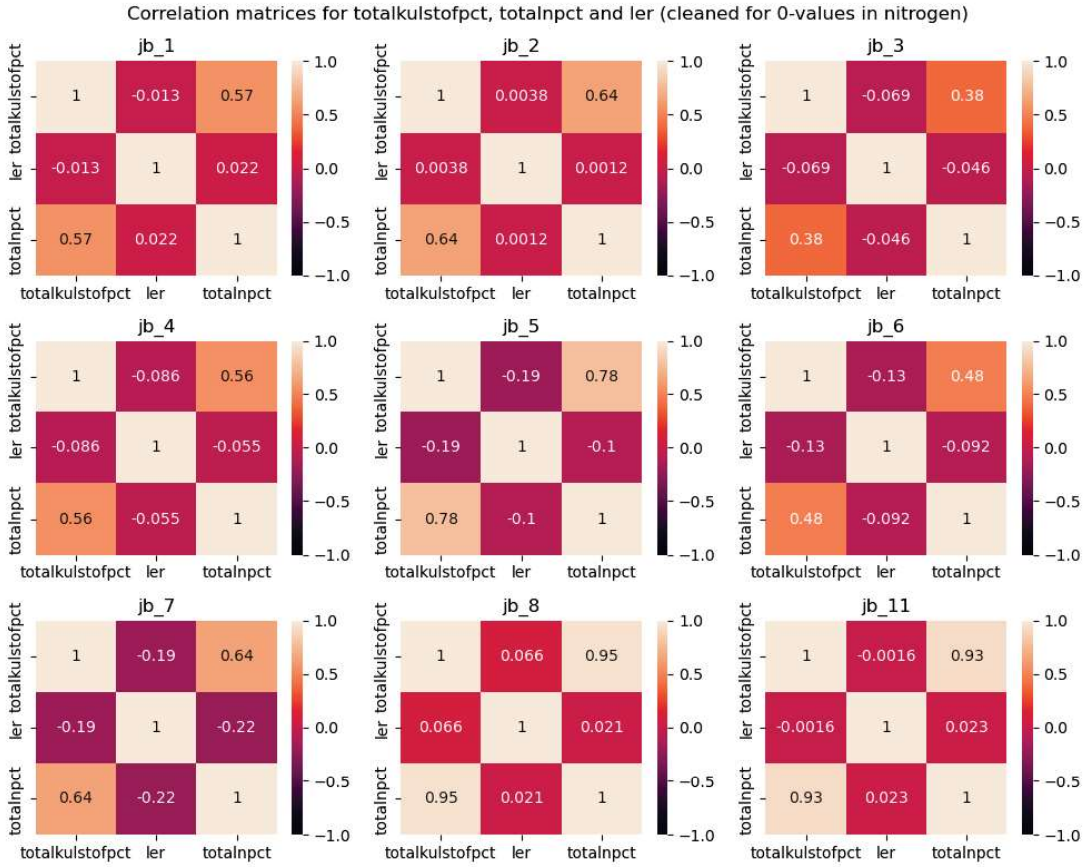
[ ]:
```
fig, axs = plt.subplots(3, 3, figsize = (10,8))

fig.suptitle('Correlation matrices for totalkulstofpct, totalnpct and ler')

for i in range(3):
    for j in range(3):
```

```
        idx = i * 3 + j + 1
        if idx <= 8:
            dataframe_name = f'jb_{idx}_df'
            title = f'jb_{idx}'
        else:
            dataframe_name = 'jb_11_df'
            title = 'jb_11'

        sns.heatmap(jb_dataframes[dataframe_name][['totalkulstofpct', 'ler',␣
↪'totalnpct']].corr(), ax =axs[i, j], vmin = -1, annot = True)
        axs[i,j].set_title(title)

plt.tight_layout()

plt.show()
```



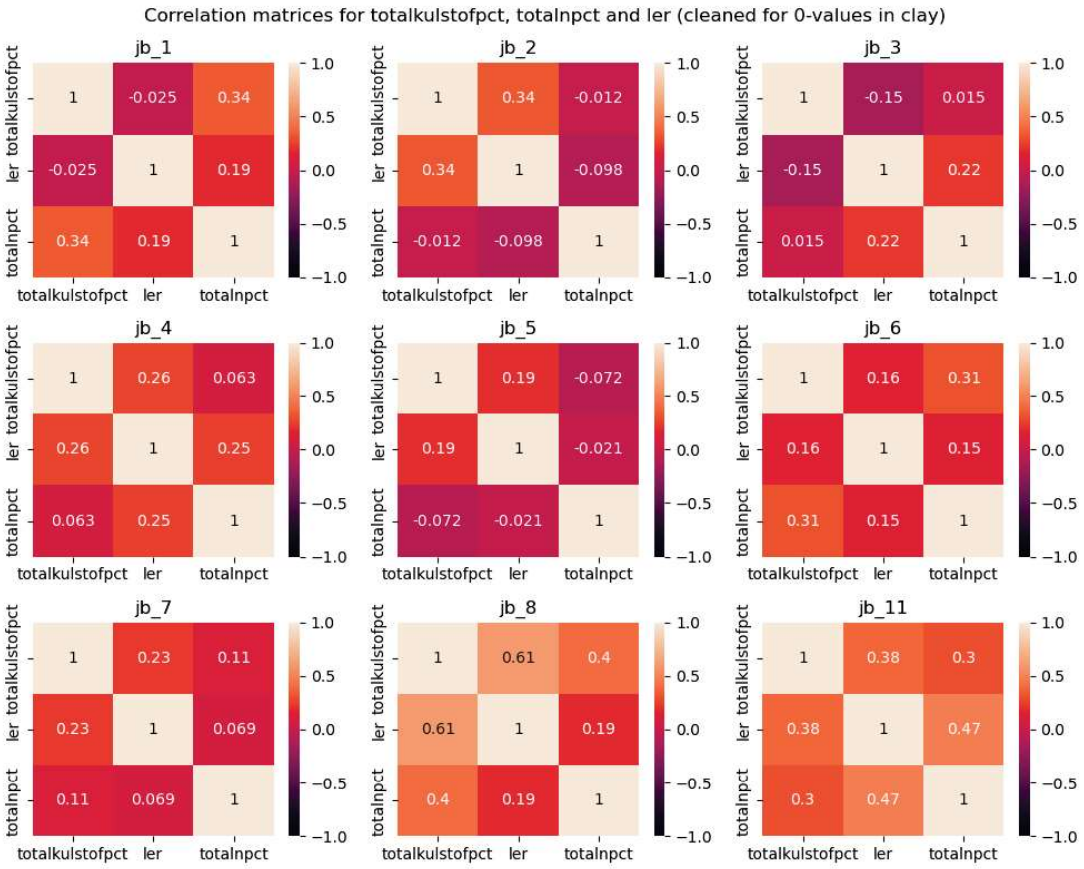
[ ]:
```
fig, axs = plt.subplots(3, 3, figsize = (10,8))
```

```
fig.suptitle('Correlation matrices for totalkulstofpct, totalnpct and ler␣
 ↪(cleaned for 0-values in nitrogen)')

for i in range(3):
    for j in range(3):
        idx = i * 3 + j + 1
        if idx <= 8:
            dataframe_name = f'jb_{idx}_df_nitrogen'
            title = f'jb_{idx}'
        else:
            dataframe_name = 'jb_11_df_nitrogen'
            title = 'jb_11'

        sns.heatmap(nitrogen_dataframes[dataframe_name][['totalkulstofpct',␣
 ↪'ler', 'totalnpct']].corr(), ax =axs[i, j], vmin = -1, annot = True)
        axs[i,j].set_title(title)

plt.tight_layout()

plt.show()
```



We found close to the same correlation between 'totalkulstofpct' and 'clay', while an increase in the correlation between 'totalkulstofpct' and 'totalnpct' which were to be expected. specifically for the soil sample numbers 8 and 5 we see a significant increase.

[ ]:
```
fig, axs = plt.subplots(3, 3, figsize = (10,8))

fig.suptitle('Correlation matrices for totalkulstofpct, totalnpct and ler␣
 ↪(cleaned for 0-values in clay)')

for i in range(3):
    for j in range(3):
        idx = i * 3 + j + 1
        if idx <= 8:
            dataframe_name = f'jb_{idx}_df_clay'
            title = f'jb_{idx}'
        else:
            dataframe_name = 'jb_11_df_clay'
            title = 'jb_11'

        sns.heatmap(clay_dataframes[dataframe_name][['totalkulstofpct', 'ler',␣
 ↪'totalnpct']].corr(), ax =axs[i, j], vmin = -1, annot = True)
        axs[i,j].set_title(title)

plt.tight_layout()

plt.show()
```



For the correlation matrices for the dataframes cleaned for 0-values in clay, we found a significant increase in the correlation between 'totalkulstofpct' and 'ler', while we still observe some correlation between 'totalnpct' and 'totalkulstofpct'.

2.0.4 Summary of regression analysis for each subset of the dataset

[ ]:
```
result_dataframes = {
    'result_jb_1': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
↪C(saedskifte)', data = jb_dataframes['jb_1_df']).fit(),
    'result_jb_2': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
↪C(saedskifte)', data = jb_dataframes['jb_2_df']).fit(),
    'result_jb_3': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
↪C(saedskifte)', data = jb_dataframes['jb_3_df']).fit(),
    'result_jb_4': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
↪C(saedskifte)', data = jb_dataframes['jb_4_df']).fit(),
    'result_jb_5': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
↪C(saedskifte)', data = jb_dataframes['jb_5_df']).fit(),
    'result_jb_6': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
↪C(saedskifte)', data = jb_dataframes['jb_6_df']).fit(),
```

```
    'result_jb_7': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
 ↪C(saedskifte)', data = jb_dataframes['jb_7_df']).fit(),
    'result_jb_8': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
 ↪C(saedskifte)', data = jb_dataframes['jb_8_df']).fit(),
    'result_jb_11': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
 ↪C(saedskifte)', data = jb_dataframes['jb_11_df']).fit()
}

result_dataframes_nitrogen = {
    'result_jb_1_nitrogen': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
 ↪C(saedskifte)', data = nitrogen_dataframes['jb_1_df']).fit(),
    'result_jb_2_nitrogen': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
 ↪C(saedskifte)', data = nitrogen_dataframes['jb_2_df']).fit(),
    'result_jb_3_nitrogen': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
 ↪C(saedskifte)', data = nitrogen_dataframes['jb_3_df']).fit(),
    'result_jb_4_nitrogen': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
 ↪C(saedskifte)', data = nitrogen_dataframes['jb_4_df']).fit(),
    'result_jb_5_nitrogen': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
 ↪C(saedskifte)', data = nitrogen_dataframes['jb_5_df']).fit(),
    'result_jb_6_nitrogen': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
 ↪C(saedskifte)', data = nitrogen_dataframes['jb_6_df']).fit(),
    'result_jb_7_nitrogen': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
 ↪C(saedskifte)', data = nitrogen_dataframes['jb_7_df']).fit(),
    'result_jb_8_nitrogen': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
 ↪C(saedskifte)', data = nitrogen_dataframes['jb_8_df']).fit(),
    'result_jb_11_nitrogen': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
 ↪C(saedskifte)', data = nitrogen_dataframes['jb_11_df']).fit()
}

result_dataframes_clay = {
    'result_jb_1_clay': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
 ↪C(saedskifte)', data = clay_dataframes['jb_1_df']).fit(),
    'result_jb_2_clay': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
 ↪C(saedskifte)', data = clay_dataframes['jb_2_df']).fit(),
    'result_jb_3_clay': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
 ↪C(saedskifte)', data = clay_dataframes['jb_3_df']).fit(),
    'result_jb_4_clay': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
 ↪C(saedskifte)', data = clay_dataframes['jb_4_df']).fit(),
    'result_jb_5_clay': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
 ↪C(saedskifte)', data = clay_dataframes['jb_5_df']).fit(),
    'result_jb_6_clay': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
 ↪C(saedskifte)', data = clay_dataframes['jb_6_df']).fit(),
    'result_jb_7_clay': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
 ↪C(saedskifte)', data = clay_dataframes['jb_7_df']).fit(),
    'result_jb_8_clay': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
 ↪C(saedskifte)', data = clay_dataframes['jb_8_df']).fit(),
```

```
    'result_jb_11_clay': ols(formula= 'totalkulstofpct ~ totalnpct + ler +␣
 ↪C(saedskifte)', data = clay_dataframes['jb_11_df']).fit()
}
```

Summary of regression analysis for subsets based soil sample numbers

[ ]:
```
summary_df = pd.DataFrame(columns=['model_jb_1', 'model_jb_2', 'model_jb_3',␣
 ↪'model_jb_4', 'model_jb_5',
                                    'model_jb_6', 'model_jb_7', 'model_jb_8',␣
 ↪'model_jb_11'])

for i in range(9):

    if i <= 7:
        result_name = f'result_jb_{i+1}'
        model_name = f'model_jb_{i+1}'
        coefficients = result_dataframes[result_name].params
        p_values = result_dataframes[result_name].pvalues
        obs = result_dataframes[result_name].nobs
        r_squared = result_dataframes[result_name].rsquared
        new_model_name = ' '.join([f'{model_name}, Observationer: {obs}, R^2:␣
 ↪{r_squared}'])
    else:
        result_name = 'result_jb_11'
        model_name = 'model_jb_11'
        coefficients = result_dataframes[result_name].params
        p_values = result_dataframes[result_name].pvalues
        obs = result_dataframes[result_name].nobs
        r_squared = result_dataframes[result_name].rsquared
        new_model_name = ' '.join([f'{model_name}, Observationer: {obs}, R^2:␣
 ↪{r_squared}'])
    for j in range(len(result_dataframes[result_name].params.index.tolist())):
        coefficient_string = ' '.join([result_dataframes[result_name].params.
 ↪index.tolist()[j], f'{result_dataframes[result_name].params.tolist()[j]}'])
        p_value_string = ' '.join([f'P-value: {result_dataframes[result_name].
 ↪pvalues.tolist()[j]}'])
        result_string = coefficient_string + ', ' + p_value_string
        summary_df.loc[j, model_name] = result_string
    summary_df.rename(columns = {model_name: new_model_name}, inplace = True)
    print(tabulate(summary_df.iloc[:, i:i+1], headers = 'keys', tablefmt =␣
 ↪'outline'))
```

```
+----+-----------------------------------------------------------------------+
|    | model_jb_1, Observationer: 8976.0, R^2: 0.28355449517607456           |
+====+=======================================================================+
|  0 | Intercept 1.4863117530265673, P-value: 0.0                            |
|  1 | C(saedskifte)[T.1] 0.1715082550762293, P-value: 8.8661911084699e-08   |
```

2	C(saedskifte)[T.2] -0.23637445432876025, P-value: 7.774109424688644e-09
3	C(saedskifte)[T.3] 0.24818638691859368, P-value: 0.03770843219293992
4	C(saedskifte)[T.4] 0.0843661059280689, P-value: 0.6085051906596008
5	totalnpct 4.836172614353762, P-value: 0.0
6	ler 0.14974022745963542, P-value: 3.7235187095727783e-37

	model_jb_2, Observationer: 8910.0, R^2: 0.3964756557302559
0	Intercept 1.094900820945586, P-value: 0.0
1	C(saedskifte)[T.1] 0.11455493023161979, P-value: 4.13082861003708e-06
2	C(saedskifte)[T.2] -0.13118209138857237, P-value: 3.751206422425294e-05
3	C(saedskifte)[T.3] -0.07342487579208413, P-value: 0.2654564643240593
4	C(saedskifte)[T.4] -0.22812985425453156, P-value: 0.025242397121725265
5	totalnpct 5.293689705817548, P-value: 0.0
6	ler 0.24825335861137024, P-value: 1.5012599442699802e-79

	model_jb_3, Observationer: 1986.0, R^2: 0.14018207099031754
0	Intercept 1.9050114291356621, P-value: 1.6499218845872907e-149
1	C(saedskifte)[T.1] 0.2687174053104068, P-value: 0.00012640358671012368
2	C(saedskifte)[T.2] -0.3832698789483505, P-value: 4.238799202712402e-05
3	C(saedskifte)[T.3] -0.03409156527490367, P-value: 0.8219570865769493
4	C(saedskifte)[T.4] -0.43016014109175954, P-value: 0.002392856598351835
5	totalnpct 1.966971181308046, P-value: 3.0738980246106818e-47
6	ler 0.019089583327976962, P-value: 0.052382366374209614

	model_jb_4, Observationer: 8032.0, R^2: 0.26157612748198167
0	Intercept 1.573530613370959, P-value: 0.0
1	C(saedskifte)[T.1] 0.04845410509928605, P-value: 0.1159790183279882
2	C(saedskifte)[T.2] -0.26045605992210963, P-value: 4.349057940037583e-09
3	C(saedskifte)[T.3] 0.3172588572142929, P-value: 3.9282012442825586e-05
4	C(saedskifte)[T.4] -0.3949687314148669, P-value: 2.751155177797543e-10
5	totalnpct 3.883862968546919, P-value: 0.0
6	ler 0.045344565864209846, P-value: 5.111346725366799e-23

	model_jb_5, Observationer: 316.0, R^2: 0.28348540564935665
0	Intercept 0.9966494119275113, P-value: 3.9780251047757434e-11
1	C(saedskifte)[T.1] -0.02215726795757185, P-value: 0.8761734014747733
2	C(saedskifte)[T.2] -0.07811062051471966, P-value: 0.663450584780235
3	C(saedskifte)[T.3] 0.629426472954912, P-value: 0.009038496681473588
4	C(saedskifte)[T.4] -0.6024041307299961, P-value: 0.0004692859556577839
5	totalnpct 6.155569650109041, P-value: 1.2600241421515765e-20

6	ler 0.05412423064285869, P-value: 8.762780228154688e-06

	model_jb_6, Observationer: 4793.0, R^2: 0.22611176726899285
0	Intercept 1.3408727405396648, P-value: 0.0
1	C(saedskifte)[T.1] 0.327878158896896, P-value: 2.872101443830077e-24
2	C(saedskifte)[T.2] -0.06994100109251147, P-value: 0.2728057695406669
3	C(saedskifte)[T.3] 0.31327045501362466, P-value: 2.9977242757657676e-07
4	C(saedskifte)[T.4] -0.18416843196109453, P-value: 7.100819615271297e-07
5	totalnpct 2.734219710888433, P-value: 1.1737785463423671e-178
6	ler 0.014803391494558628, P-value: 1.9462503597304954e-09

	model_jb_7, Observationer: 384.0, R^2: 0.19910328269481903
0	Intercept 1.304801472244744, P-value: 2.8756944661382615e-17
1	C(saedskifte)[T.1] -0.016899140878324365, P-value: 0.9111130935928148
2	C(saedskifte)[T.2] -0.6762313621928804, P-value: 0.5455114501810883
3	C(saedskifte)[T.3] 0.37241909146411434, P-value: 0.1316026613492768
4	C(saedskifte)[T.4] 0.04523164027658266, P-value: 0.8109304908769914
5	totalnpct 4.266162876001863, P-value: 3.2429426780424197e-16
6	ler 0.010959428045149615, P-value: 0.26010976898054644

	model_jb_8, Observationer: 167.0, R^2: 0.4723434712085841
0	Intercept 1.3417966025650798, P-value: 3.5183206558231122e-06
1	C(saedskifte)[T.1] 0.0511159811536708, P-value: 0.8376656241925826
2	C(saedskifte)[T.3] 1.7268552631902125, P-value: 0.03527492213822279
3	C(saedskifte)[T.4] -1.1150063318100352, P-value: 0.1949227444924721
4	totalnpct 7.868195681390582, P-value: 3.6566215244680873e-22
5	ler 0.07164378183697151, P-value: 0.006887866273955972
6	nan

	model_jb_11, Observationer: 3188.0, R^2: 0.7073822143577857
0	Intercept 2.7031779702563594, P-value: 5.290384832844428e-101
1	C(saedskifte)[T.1] -0.2743102872033145, P-value: 0.03489092150550566
2	C(saedskifte)[T.2] -0.5867075992521263, P-value: 0.027748533394958814
3	C(saedskifte)[T.3] 0.18056362281730184, P-value: 0.4389545840848894
4	C(saedskifte)[T.4] -0.8371797374313938, P-value: 0.002973870359230042
5	totalnpct 8.657018678100002, P-value: 0.0
6	ler 0.16585296872838584, P-value: 4.836705470504674e-63

Summary of regression analysis for subsets based soil sample numbers cleaned for 0-values in nitrogen

[ ]:
```
summary_df_nitrogen = pd.DataFrame(columns=['model_jb_1_nitrogen',␣
↪'model_jb_2_nitrogen', 'model_jb_3_nitrogen', 'model_jb_4_nitrogen',␣
↪'model_jb_5_nitrogen',
                                    'model_jb_6_nitrogen',␣
↪'model_jb_7_nitrogen', 'model_jb_8_nitrogen', 'model_jb_11_nitrogen'])

for i in range(9):

    if i <= 7:
        result_name = f'result_jb_{i+1}_nitrogen'
        model_name = f'model_jb_{i+1}_nitrogen'
        coefficients = result_dataframes_nitrogen[result_name].params
        p_values = result_dataframes_nitrogen[result_name].pvalues
        obs = result_dataframes_nitrogen[result_name].nobs
        r_squared = result_dataframes_nitrogen[result_name].rsquared
        new_model_name = ' '.join([f'{model_name}, Observationer: {obs}, R^2:␣
↪{r_squared}'])
    else:
        result_name = 'result_jb_11_nitrogen'
        model_name = 'model_jb_11_nitrogen'
        coefficients = result_dataframes_nitrogen[result_name].params
        p_values = result_dataframes_nitrogen[result_name].pvalues
        obs = result_dataframes_nitrogen[result_name].nobs
        r_squared = result_dataframes_nitrogen[result_name].rsquared
        new_model_name = ' '.join([f'{model_name}, Observationer: {obs}, R^2:␣
↪{r_squared}'])
    for j in range(len(result_dataframes_nitrogen[result_name].params.index.
↪tolist())):
        coefficient_string = ' '.join([result_dataframes_nitrogen[result_name].
↪params.index.tolist()[j], f'{result_dataframes_nitrogen[result_name].params.
↪tolist()[j]}'])
        p_value_string = ' '.join([f'P-value:␣
↪{result_dataframes_nitrogen[result_name].pvalues.tolist()[j]}'])
        result_string = coefficient_string + ', ' + p_value_string
        summary_df_nitrogen.loc[j, model_name] = result_string
    summary_df_nitrogen.rename(columns = {model_name: new_model_name}, inplace␣
↪= True)
    print(tabulate(summary_df_nitrogen.iloc[:, i:i+1], headers = 'keys',␣
↪tablefmt = 'outline'))
```

```
+----+-----------------------------------------------------------------------+
|    | model_jb_1_nitrogen, Observationer: 8560.0, R^2: 0.33397765795399015   |
+====+=======================================================================+
|  0 | Intercept 1.3593902708765766, P-value: 0.0                            |
|  1 | C(saedskifte)[T.1] 0.17017375581763122, P-value: 5.46520372614469e-08  |
```

2	C(saedskifte)[T.2] -0.19296797034298685, P-value: 1.054563589051166e-06
3	C(saedskifte)[T.3] 0.14302520878614755, P-value: 0.2640100866466002
4	C(saedskifte)[T.4] -0.11026906464540145, P-value: 0.6301116781786543
5	totalnpct 5.2602628531996025, P-value: 0.0
6	ler -0.04814178678234086, P-value: 0.018390596852879748

	model_jb_2_nitrogen, Observationer: 8811.0, R^2: 0.41156525854155657

0	Intercept 1.0605822363553028, P-value: 0.0
1	C(saedskifte)[T.1] 0.12976324382576684, P-value: 9.958846633567715e-08
2	C(saedskifte)[T.2] -0.10827126843315614, P-value: 0.00048585846142352546
3	C(saedskifte)[T.3] -0.03482659936574398, P-value: 0.590811313909684
4	C(saedskifte)[T.4] -0.0725361994127321, P-value: 0.49839195062637953
5	totalnpct 5.362323797576519, P-value: 0.0
6	ler 0.027405096588230874, P-value: 0.6862525248154421

	model_jb_3_nitrogen, Observationer: 1784.0, R^2: 0.18482910755127735

0	Intercept 1.9092432686036251, P-value: 7.750255803713383e-155
1	C(saedskifte)[T.1] 0.13781547051374798, P-value: 0.04201871490630277
2	C(saedskifte)[T.2] -0.47035811530089855, P-value: 1.0768630206254203e-07
3	C(saedskifte)[T.3] -0.14897909857800376, P-value: 0.3422920147714228
4	C(saedskifte)[T.4] -0.5169620287580919, P-value: 0.004288957809952472
5	totalnpct 2.2078196818326488, P-value: 3.408805534522517e-63
6	ler -0.0057247622503996546, P-value: 0.6540320660357269

	model_jb_4_nitrogen, Observationer: 7465.0, R^2: 0.32046209090648237
0	Intercept 1.506285628627528, P-value: 0.0
1	C(saedskifte)[T.1] 0.04509760511694164, P-value: 0.12703358371607815
2	C(saedskifte)[T.2] -0.2442378277585288, P-value: 5.9838121941790735e-09
3	C(saedskifte)[T.3] 0.04888177174821978, P-value: 0.5472943396151493
4	C(saedskifte)[T.4] -0.4263360561554413, P-value: 6.48898056498321e-08
5	totalnpct 4.163754801540026, P-value: 0.0
6	ler -0.009419369627704975, P-value: 0.13789494102580024

	model_jb_5_nitrogen, Observationer: 227.0, R^2: 0.6260127383724611
0	Intercept 0.31055874630663843, P-value: 0.02096422593151954
1	C(saedskifte)[T.1] 0.01596795840041241, P-value: 0.8801932742605703
2	C(saedskifte)[T.2] 0.09920039405055299, P-value: 0.4415484633564025
3	C(saedskifte)[T.3] -0.14494380269396873, P-value: 0.7200470307401818
4	C(saedskifte)[T.4] -0.05991460430285955, P-value: 0.7872618029297469
5	totalnpct 9.522252299121213, P-value: 3.3634001815886583e-46
6	ler -0.016317578594286026, P-value: 0.29278168827150514

	model_jb_6_nitrogen, Observationer: 3374.0, R^2: 0.27624266019992216
0	Intercept 1.1839633654910842, P-value: 1.3586723737430726e-197
1	C(saedskifte)[T.1] 0.3480569473215497, P-value: 1.4157589047089492e-23
2	C(saedskifte)[T.2] -0.022275690708626852, P-value: 0.7237476395481203
3	C(saedskifte)[T.3] 0.10837342016915995, P-value: 0.20732186707179293
4	C(saedskifte)[T.4] -0.36370797519917736, P-value: 7.165902926853819e-09
5	totalnpct 3.31594855505381, P-value: 7.646350525426057e-182
6	ler 0.01694112750764823, P-value: 1.4130148206384247e-05

	model_jb_7_nitrogen, Observationer: 249.0, R^2: 0.42106554439326016
0	Intercept 0.8735735405409863, P-value: 1.6262453748908574e-06
1	C(saedskifte)[T.1] -0.17251634420154005, P-value: 0.2447921829922338
2	C(saedskifte)[T.2] -0.42169363476873006, P-value: 0.647619127555574
3	C(saedskifte)[T.3] -0.5198104306879493, P-value: 0.047500979868143
4	C(saedskifte)[T.4] -0.2579861777294351, P-value: 0.33341390813161553
5	totalnpct 6.686576633256807, P-value: 7.566567152159108e-28
6	ler -0.011936067677458462, P-value: 0.341661265118388

	model_jb_8_nitrogen, Observationer: 151.0, R^2: 0.9052273177859319

```
+====+=============================================================================+
|  0 | Intercept -0.007345732717078297, P-value: 0.9544678517348559                |
|  1 | C(saedskifte)[T.1] -0.04692907977753841, P-value: 0.6524477151887468        |
|  2 | C(saedskifte)[T.3] 1.150245812614167, P-value: 0.06665927539873943          |
|  3 | C(saedskifte)[T.4] -1.6839794371032895, P-value: 0.001988670038790679       |
|  4 | totalnpct 11.293429359139257, P-value: 1.5977059701781792e-72                |
|  5 | ler 0.06566500213139992, P-value: 0.0012498700792667194                      |
|  6 | nan                                                                          |
+----+------------------------------------------------------------------------------+
+----+-----------------------------------------------------------------------------+
|    | model_jb_11_nitrogen, Observationer: 2404.0, R^2: 0.8575716032282938        |
+====+=============================================================================+
|  0 | Intercept 1.2270412991899646, P-value: 3.4507412712224494e-26                |
|  1 | C(saedskifte)[T.1] -0.06462442046015007, P-value: 0.5778188050250364         |
|  2 | C(saedskifte)[T.2] 0.27125790461142796, P-value: 0.22798309576188264         |
|  3 | C(saedskifte)[T.3] -0.20045167776573325, P-value: 0.41972459559388564        |
|  4 | C(saedskifte)[T.4] -0.8700562832125484, P-value: 0.009637008802659603        |
|  5 | totalnpct 10.05048851849164, P-value: 0.0                                    |
|  6 | ler -0.014015190805243254, P-value: 0.2006650620421946                       |
+----+------------------------------------------------------------------------------+
```

Summary of regression analysis for subsets based soil sample numbers cleaned for 0-values in clay

[ ]:
```
summary_df_clay = pd.DataFrame(columns=['model_jb_1_clay', 'model_jb_2_clay',␣
↪'model_jb_3_clay', 'model_jb_4_clay', 'model_jb_5_clay',
                                    'model_jb_6_clay', 'model_jb_7_clay',␣
↪'model_jb_8_clay', 'model_jb_11_clay'])

for i in range(9):

    if i <= 7:
        result_name = f'result_jb_{i+1}_clay'
        model_name = f'model_jb_{i+1}_clay'
        coefficients = result_dataframes_clay[result_name].params
        p_values = result_dataframes_clay[result_name].pvalues
        obs = result_dataframes_clay[result_name].nobs
        r_squared = result_dataframes_clay[result_name].rsquared
        new_model_name = ' '.join([f'{model_name}, Observationer: {obs}, R^2:␣
↪{r_squared}'])
    else:
        result_name = 'result_jb_11_clay'
        model_name = 'model_jb_11_clay'
        coefficients = result_dataframes_clay[result_name].params
        p_values = result_dataframes_clay[result_name].pvalues
        obs = result_dataframes_clay[result_name].nobs
        r_squared = result_dataframes_clay[result_name].rsquared
```

```
        new_model_name = ' '.join([f'{model_name}, Observationer: {obs}, R^2:␣
↪{r_squared}'])
    for j in range(len(result_dataframes_clay[result_name].params.index.
↪tolist())):
        coefficient_string = ' '.join([result_dataframes_clay[result_name].
↪params.index.tolist()[j], f'{result_dataframes_clay[result_name].params.
↪tolist()[j]}'])
        p_value_string = ' '.join([f'P-value:␣
↪{result_dataframes_clay[result_name].pvalues.tolist()[j]}'])
        result_string = coefficient_string + ', ' + p_value_string
        summary_df_clay.loc[j, model_name] = result_string
    summary_df_clay.rename(columns = {model_name: new_model_name}, inplace =␣
↪True)
    print(tabulate(summary_df_clay.iloc[:, i:i+1], headers = 'keys', tablefmt =␣
↪'outline'))
```

```
+----+-----------------------------------------------------------------------+
|    | model_jb_1_clay, Observationer: 344.0, R^2: 0.13672854298730586       |
+====+=======================================================================+
|  0 | Intercept 3.065458020076928, P-value: 5.09138853496237e-21            |
|  1 | C(saedskifte)[T.1] 0.370179931561967, P-value: 0.18432182362262445    |
|  2 | C(saedskifte)[T.2] -0.4151319464477906, P-value: 0.40800103876570015  |
|  3 | C(saedskifte)[T.3] 0.2963910125748254, P-value: 0.5711959218975626    |
|  4 | C(saedskifte)[T.4] -0.26215952827436234, P-value: 0.4513280414730083  |
|  5 | totalnpct 4.715893255218717, P-value: 1.0412790534380415e-10           |
|  6 | ler -0.0769736127072111, P-value: 0.08727762329918824                 |
+----+-----------------------------------------------------------------------+
+----+-----------------------------------------------------------------------+
|    | model_jb_2_clay, Observationer: 73.0, R^2: 0.1745645991455823         |
+====+=======================================================================+
|  0 | Intercept 1.5041176952509785, P-value: 0.015545247727516635           |
|  1 | C(saedskifte)[T.1] -1.07511749297766, P-value: 0.27645868942261054    |
|  2 | C(saedskifte)[T.2] -2.1760432292243617, P-value: 0.392639883920366    |
|  3 | C(saedskifte)[T.3] -1.5478939451777605, P-value: 0.20635159926985913  |
|  4 | C(saedskifte)[T.4] -1.480699841880806, P-value: 0.05309648490044922   |
|  5 | totalnpct 3.9605355709086774, P-value: 0.48887723781117276            |
|  6 | ler 0.28608653154526076, P-value: 0.0005705959170805898               |
+----+-----------------------------------------------------------------------+
+----+-----------------------------------------------------------------------+
|    | model_jb_3_clay, Observationer: 279.0, R^2: 0.24546917630183573       |
+====+=======================================================================+
|  0 | Intercept 2.0734155014451474, P-value: 6.725342262791716e-14           |
|  1 | C(saedskifte)[T.1] 1.990395775314027, P-value: 4.1534282025041565e-13 |
|  2 | C(saedskifte)[T.2] 2.6025644173754445, P-value: 0.0021295975284648426 |
|  3 | C(saedskifte)[T.3] 0.4225997698435538, P-value: 0.28252265767578       |
|  4 | C(saedskifte)[T.4] -0.19163970735413938, P-value: 0.3379146294050652   |
|  5 | totalnpct 1.41248500481783, P-value: 0.15307643263027332              |
```

6	ler -0.026213291537781985, P-value: 0.29666781205548304

	model_jb_4_clay, Observationer: 817.0, R^2: 0.10322485619296973
0	Intercept 1.204624368297311, P-value: 4.747523570164059e-14
1	C(saedskifte)[T.1] -0.011426584026043575, P-value: 0.954133036084505
2	C(saedskifte)[T.2] -0.04899155078331103, P-value: 0.9123733142653608
3	C(saedskifte)[T.3] 0.7815106997143032, P-value: 0.0002050095898604758
4	C(saedskifte)[T.4] -0.3411078186346291, P-value: 0.002605992877208437
5	totalnpct 0.5741963256029892, P-value: 0.2885511659547555
6	ler 0.09474928463625609, P-value: 6.854822706739855e-11

	model_jb_5_clay, Observationer: 114.0, R^2: 0.09530404521880065
0	Intercept 1.0424583399650837, P-value: 0.0049536033905288185
1	C(saedskifte)[T.1] 0.24151623405805256, P-value: 0.6098523601917609
2	C(saedskifte)[T.3] 0.6756111703474792, P-value: 0.050391926380642264
3	C(saedskifte)[T.4] -0.19899185609276998, P-value: 0.41823226618583786
4	totalnpct 0.1710659168897563, P-value: 0.9095748207046341
5	ler 0.05500752786710647, P-value: 0.038100726779034144
6	nan

	model_jb_6_clay, Observationer: 1832.0, R^2: 0.1363920917444238
0	Intercept 1.1302363589851954, P-value: 6.6050879754169e-48
1	C(saedskifte)[T.1] 0.29600044489150645, P-value: 0.011973966253233362
2	C(saedskifte)[T.2] 0.0011449825736743469, P-value: 0.9977977405763241
3	C(saedskifte)[T.3] 0.2869655269251563, P-value: 0.001972058815485921
4	C(saedskifte)[T.4] -0.23913997330322725, P-value: 4.803847651098249e-07
5	totalnpct 2.4227119304545814, P-value: 9.059367181127456e-42
6	ler 0.03359857122596042, P-value: 2.3938141355039577e-09

	model_jb_7_clay, Observationer: 159.0, R^2: 0.11809776756189405
0	Intercept 0.5861619706431319, P-value: 0.1099623013727074
1	C(saedskifte)[T.1] 1.1171549418306117, P-value: 0.013846135501671878
2	C(saedskifte)[T.3] 0.8586660697657222, P-value: 0.047480930910876476
3	C(saedskifte)[T.4] 0.2266861710066088, P-value: 0.28110977194042147
4	totalnpct 1.1270015292756224, P-value: 0.321463359852276
5	ler 0.055735886940049335, P-value: 0.016933641815889047
6	nan

	model_jb_8_clay, Observationer: 13.0, R^2: 0.8563079125248736

```
+====+==========================================================================+
|  0 | Intercept 0.47485386117549033, P-value: 0.7193068217309974               |
|  1 | C(saedskifte)[T.1] 3.471589159380175, P-value: 0.02556162465027137       |
|  2 | C(saedskifte)[T.3] 2.407787285522739, P-value: 0.025004275932747577      |
|  3 | C(saedskifte)[T.4] 0.1492786562864448, P-value: 0.889112361558689        |
|  4 | totalnpct 2.3373356159492005, P-value: 0.3842807546979279                |
|  5 | ler 0.07759622231939438, P-value: 0.11119322828087146                    |
|  6 | nan                                                                      |
+----+--------------------------------------------------------------------------+
+----+--------------------------------------------------------------------------+
|    | model_jb_11_clay, Observationer: 974.0, R^2: 0.16954812549805143         |
+====+==========================================================================+
|  0 | Intercept 4.4679634341118195, P-value: 6.628267489858277e-85             |
|  1 | C(saedskifte)[T.1] 0.3890934869145448, P-value: 0.11978236282932632      |
|  2 | C(saedskifte)[T.2] 0.18498726777722593, P-value: 0.7498713002641002      |
|  3 | C(saedskifte)[T.3] 0.12411381412226527, P-value: 0.6856136235858649      |
|  4 | C(saedskifte)[T.4] -0.5673830156425212, P-value: 0.07093339869066573     |
|  5 | totalnpct 2.0604292806415003, P-value: 3.9129356321157545e-07            |
|  6 | ler 0.13646673298722614, P-value: 4.790907369589068e-19                  |
+----+--------------------------------------------------------------------------+
```

2.0.5 Summary of total number of variables, means and standard deviation for cleaned sub-datasets and each crop rotation number

Number of observations based on the crop rotation number

[ ]:
```
saed_df = (
    gdf
    .groupby('saedskifte')
    .size()
)

print(saed_df)
```

```
saedskifte
0     9137
1    19864
2     4728
3     1108
4     1916
dtype: int64
```

Means for the whole dataset, and for the cleaned sub-datasets

[ ]:
```
for i in range(9):

    if i <= 7:
        model_name = f'jb_{i+1}_df'
    else:
```

```
        model_name = 'jb_11_df'
    mean_number = jb_dataframes[model_name]['totalkulstofpct'].mean()
    mean_string = ' '.join([f'Means for totalkulstofpct for {model_name}:␣
↪{mean_number}'])
    print(mean_string)
```

```
Means for totalkulstofpct for jb_1_df: 2.54803795651453
Means for totalkulstofpct for jb_2_df: 2.2324435570172003
Means for totalkulstofpct for jb_3_df: 2.3923761680601423
Means for totalkulstofpct for jb_4_df: 2.4305561135150002
Means for totalkulstofpct for jb_5_df: 1.9545282602296141
Means for totalkulstofpct for jb_6_df: 1.9159157067379504
Means for totalkulstofpct for jb_7_df: 1.9888178294573644
Means for totalkulstofpct for jb_8_df: 3.9573346330594625
Means for totalkulstofpct for jb_11_df: 7.102352353301626
```

[ ]:
```
for i in range(9):

    if i <= 7:
        model_name = f'jb_{i+1}_df_nitrogen'
    else:
        model_name = 'jb_11_df_nitrogen'
    mean_number = nitrogen_dataframes[model_name]['totalkulstofpct'].mean()
    mean_string = ' '.join([f'Means for totalkulstofpct for {model_name}:␣
↪{mean_number}'])
    print(mean_string)
```

```
Means for totalkulstofpct for jb_1_df_nitrogen: 2.5265278852423387
Means for totalkulstofpct for jb_2_df_nitrogen: 2.2252901368004587
Means for totalkulstofpct for jb_3_df_nitrogen: 2.387043617686933
Means for totalkulstofpct for jb_4_df_nitrogen: 2.438342478294989
Means for totalkulstofpct for jb_5_df_nitrogen: 2.0093361335928694
Means for totalkulstofpct for jb_6_df_nitrogen: 2.0473685004387887
Means for totalkulstofpct for jb_7_df_nitrogen: 2.120608947417577
Means for totalkulstofpct for jb_8_df_nitrogen: 3.8379978438318187
Means for totalkulstofpct for jb_11_df_nitrogen: 7.445070908950199
```

[ ]:
```
for i in range(9):

    if i <= 7:
        model_name = f'jb_{i+1}_df_clay'
    else:
        model_name = 'jb_11_df_clay'
    mean_number = clay_dataframes[model_name]['totalkulstofpct'].mean()
    mean_string = ' '.join([f'Middelværdi af totalkulstofpct for {model_name}:␣
↪{mean_number}'])
    print(mean_string)
```

```
Middelværdi af totalkulstofpct for jb_1_df_clay: 2.95309221200649
Middelværdi af totalkulstofpct for jb_2_df_clay: 3.004603376871615
Middelværdi af totalkulstofpct for jb_3_df_clay: 2.2252288072018005
Middelværdi af totalkulstofpct for jb_4_df_clay: 2.1026093763342915
Middelværdi af totalkulstofpct for jb_5_df_clay: 1.7589147286821705
Middelværdi af totalkulstofpct for jb_6_df_clay: 1.6435944957855186
Middelværdi af totalkulstofpct for jb_7_df_clay: 1.6570820535322512
Middelværdi af totalkulstofpct for jb_8_df_clay: 3.7955635062611814
Middelværdi af totalkulstofpct for jb_11_df_clay: 6.192923212836063
```

Means and standard deviation based on the crop rotation

[ ]:
```
saedskifte_dataframes = {
    'saed_0': gdf[gdf['saedskifte'] == 0],
    'saed_1': gdf[gdf['saedskifte'] == 1],
    'saed_2': gdf[gdf['saedskifte'] == 2],
    'saed_3': gdf[gdf['saedskifte'] == 3],
    'saed_4': gdf[gdf['saedskifte'] == 4]
}

for i in range(5):
    model_name = f'saed_{i}'
    mean_number = round(saedskifte_dataframes[model_name]['totalkulstofpct'].
 ↪mean(), 3)
    std_error = round(saedskifte_dataframes[model_name]['totalkulstofpct'].
 ↪std(), 3)
    string_value = ' '.join([f'Mean for crop rotation {i}: {mean_number} and␣
 ↪standard deviation: {std_error}'])
    print(string_value)
```

```
Mean for crop rotation 0: 2.717 and standard deviation: 2.654
Mean for crop rotation 1: 2.953 and standard deviation: 2.632
Mean for crop rotation 2: 2.081 and standard deviation: 1.145
Mean for crop rotation 3: 3.162 and standard deviation: 2.72
Mean for crop rotation 4: 2.129 and standard deviation: 1.806
```