test_pandasai

May 24, 2023

```
[ ]: import pandas as pd
     import pandasai as pdai
     from pandasai.llm.openai import OpenAI
     from pandasai import PandasAI
```

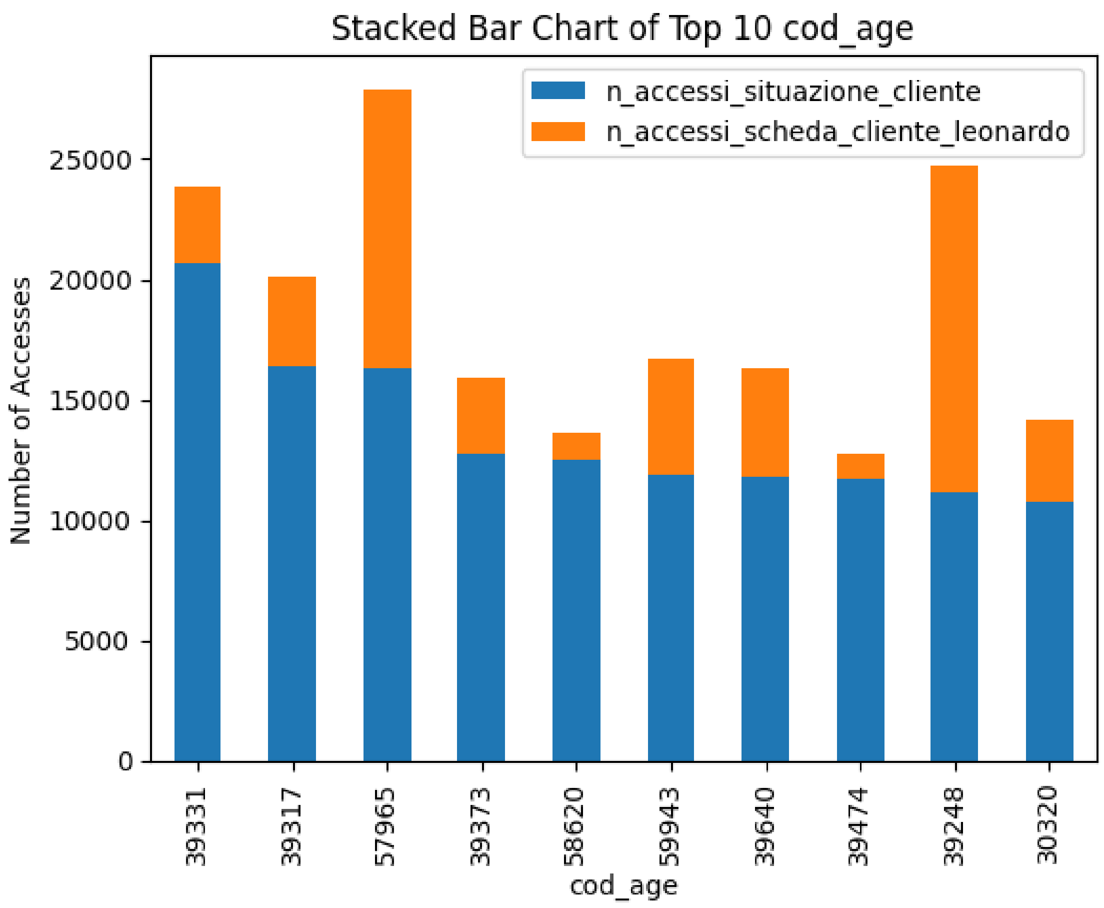
```
[ ]: path = '/mnt/c/Users/GiovanniPalazzo/Downloads/Output Apr2023.xlsx'
     data = pd.read_excel(path)
     data
```

```
[ ]:        Unnamed: 0 aux_data_run        data  cod_age data_inizio  data_fine  
     0               0   23/05/2023  01/04/2023    10650  14/03/2023        NaN  \
     1               1   23/05/2023  03/04/2023    10650  14/03/2023        NaN   
     2               2   23/05/2023  04/04/2023    10650  14/03/2023        NaN   
     3               3   23/05/2023  05/04/2023    10650  14/03/2023        NaN   
     4               4   23/05/2023  06/04/2023    10650  14/03/2023        NaN   
     …             …          …           …       …         …          …   
     36346       36346   23/05/2023  17/04/2023    72930  14/03/2023        NaN   
     36347       36347   23/05/2023  19/04/2023    72930  14/03/2023        NaN   
     36348       36348   23/05/2023  21/04/2023    72930  14/03/2023        NaN   
     36349       36349   23/05/2023  24/04/2023    72930  14/03/2023        NaN   
     36350       36350   23/05/2023  28/04/2023    72930  14/03/2023        NaN   

            n_accessi_situazione_cliente  n_accessi_scheda_cliente_leonardo  
     0                                11                                  4  
     1                                71                                 25  
     2                                76                                 30  
     3                                86                                 15  
     4                                94                                 19  
     …                               …                                 …  
     36346                             1                                  0  
     36347                             3                                  0  
     36348                             3                                  0  
     36349                             1                                  0  
     36350                             1                                  0  

     [36351 rows x 8 columns]
```

```
[ ]: llm = OpenAI(api_token="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")
```

```
[ ]: pandas_ai = PandasAI(llm)
```

```
[ ]: pandas_ai(
         data,
         "Plot the stacked bar chart of cod_age showing for each␣
      ↪n_accessi_situazione_cliente and n_accessi_scheda_cliente_leonardo,\
         using different colors for each bar. Take into account only the first 10␣
      ↪cod_age ordered by n_accessi_situazione_cliente decreasing"
     )
```



```
[ ]: 'Sure, I can help you with that! To create a stacked bar chart of cod_age, we
     need to consider two variables: n_accessi_situazione_cliente and
     n_accessi_scheda_cliente_leonardo. We will use different colors for each bar.
     However, we will only take into account the first 10 cod_age ordered by
     n_accessi_situazione_cliente decreasing.'
```

```
[ ]: tst = pandas_ai.run(data, prompt='What are the first 10 cod_age ordered by␣
     ↪n_accessi_situazione_cliente descending?')
     tst
```

```
[ ]: 'The first 10 cod_age, ordered by n_accessi_situazione_cliente in descending
     order are: 11277, 11282, 11293, 11295, 11287, 11284, 11279, 11290, 11278, and
     11288.'
```

```
[ ]: tst_2 = pandas_ai(
         data,
         "What are the first 10 cod_age ordered by n_accessi_situazione_cliente␣
     ↪descending?",
     )

     tst_2
```

```
[ ]: 'The first 10 cod_age ordered by n_accessi_situazione_cliente descending are…'
```

```
[ ]: pandas_ai.run(data, prompt='When we had the highest value of␣
     ↪n_accessi_situazione_cliente? And for n_accessi_scheda_cliente_leonardo?')
```

```
[ ]: "The highest number of accesses to the customer situation occurred on April 4th,
     2023. And the highest number of accesses to Leonardo's customer card occurred on
     April 5th, 2023."
```

```
[ ]: pandas_ai.run(data, prompt='Can you compute the grouped value by date of the␣
     ↪average of n_accessi_situazione_cliente ?')
```

```
[ ]: 'Sure! The average value of n_accessi_situazione_cliente is computed and grouped
     by date. Here are the results:\n\nOn April 1st, the average value was 11.345411.
     On April 2nd, there were no accesses recorded. On April 3rd, the average value
     was 116.258376, and so on for each day in April.'
```

```
[ ]: tst_3 = pandas_ai(
         data,
         "Return a dataframe with the grouped values by date of the average of␣
     ↪n_accessi_situazione_cliente"
     )
```

```
[ ]: tst_3
```

```
[ ]: 'The requested dataframe shows the average number of
     "n_accessi_situazione_cliente" grouped by date. Each row represents a specific
     date and its corresponding average value. The values are calculated based on the
     data provided in the original dataset.'
```

```
[ ]: pandas_ai(
         data,
         "Where is the dataframe requested in the previous question?"
     )
```

```
[ ]: 'The requested dataframe is stored in a variable called "aux_data_run" and
     contains columns such as "data", "cod_age", "data_inizio", "data_fine",
     "n_accessi_situazione_cliente", and "n_accessi_scheda_cliente_leonardo". It
     shows information about client access to certain services during a specific time
     period.'
```

```
[ ]: pandas_ai(
         data,
         "Can you provide a short description of the dataset and its columns?"
     )
```

```
[ ]: "Sure! The dataset contains information about customer access to a certain
     service. The columns include the date of the data run, the customer's age, the
     start and end dates of their access, and the number of times they accessed the
     service. There are also two columns that indicate the number of times the
     customer accessed their account through different channels."
```

```
[ ]: tst_4 = pandas_ai(
         data,
         "Generate a dataframe with the grouped values by date of the average of␣
      ↪n_accessi_situazione_cliente"
     )
     tst_4
```

```
[ ]: 'Sure! To generate a dataframe with the grouped values by date of the average of
     n_accessi_situazione_cliente, we can use pandas groupby function. The resulting
     dataframe will have the date column and the corresponding average value of
     n_accessi_situazione_cliente for each date. Would you like to see the code for
     this?'
```

```
[ ]: pandas_ai(
         data,
         "Yes please, show the code!"
     )
```

keep into account some limitations:

```
- That model is currently overloaded with other requests.
- token (more or less 4000) limits
```

```
[ ]: tst_5 = pandas_ai(
         data,
         "Show a code in order to get a dataframe with the grouped values by date of␣
     ↪the average of n_accessi_situazione_cliente. The column with the date is␣
     ↪named 'data'"
     )
     tst_5
```

```
[ ]: "To get a dataframe with the grouped values by date of the average of
n_accessi_situazione_cliente, you can use the following code:\n\n```\ndf.groupby
('data')['n_accessi_situazione_cliente'].mean()\n```\n\nThis will group the
dataframe by the 'data' column and calculate the mean of the
'n_accessi_situazione_cliente' column for each group. The result will be a new
dataframe with the date as the index and the average value as the only column."
```

I had to specify the date column, prevously it used 'data inizio'

```
[ ]: pandas_ai(
         data,
         "And the very same code computed for n_accessi_scheda_cliente_leonardo?",
     )
```

```
---------------------------------------------------------------------------
NoCodeFoundError                          Traceback (most recent call last)
Cell In[47], line 1
----> 1 pandas_ai(
      2     data,
      3     "And the very same code computed for␣
↪n_accessi_scheda_cliente_leonardo?",
      4 )

File ~/.local/lib/python3.9/site-packages/pandasai/__init__.py:184, in PandasAI
↪__call__(self, data_frame, prompt, is_conversational_answer, show_code,␣
↪anonymize_df, use_error_correction_framework)
    174 def __call__(
    175     self,
    176     data_frame: pd.DataFrame,
   (…)
    181     use_error_correction_framework: bool = True,
    182 ) -> str:
    183     """Run the LLM with the given prompt"""
--> 184     return self.run(
    185         data_frame,
    186         prompt,
    187         is_conversational_answer,
    188         show_code,
    189         anonymize_df,
    190         use_error_correction_framework,
    191     )
```

```
File ~/.local/lib/python3.9/site-packages/pandasai/__init__.py:129, in PandasAI
 ↪run(self, data_frame, prompt, is_conversational_answer, show_code,␣
 ↪anonymize_df, use_error_correction_framework)
    126 if anonymize_df:
    127     df_head = anonymize_dataframe_head(df_head)
--> 129 code = self._llm.generate_code(
    130     self._task_instruction.format(
    131         today_date=date.today(),
    132         df_head=df_head,
    133         num_rows=data_frame.shape[0],
    134         num_columns=data_frame.shape[1],
    135         rows_to_display=rows_to_display,
    136         START_CODE_TAG=START_CODE_TAG,
    137         END_CODE_TAG=END_CODE_TAG,
    138     ),
    139     prompt,
    140 )
    141 self._original_instructions = {
    142     "question": prompt,
    143     "df_head": df_head,
   (…)
    146     "rows_to_display": rows_to_display,
    147 }
    148 self.last_code_generated = code

File ~/.local/lib/python3.9/site-packages/pandasai/llm/base.py:115, in LLM.
 ↪generate_code(self, instruction, prompt)
    108 def generate_code(self, instruction: str, prompt: str) -> str:
    109     """
    110     Generate the code based on the instruction and the given prompt.
    111 
    112     Returns:
    113         str: Code
    114     """
--> 115     return self._extract_code(self.call(instruction, prompt,␣
 ↪suffix="\n\nCode:\n"))

File ~/.local/lib/python3.9/site-packages/pandasai/llm/base.py:89, in LLM.
 ↪_extract_code(self, response, separator)
     87 code = self._polish_code(code)
     88 if not self._is_python_code(code):
---> 89     raise NoCodeFoundError("No code found in the response")
     91 return code

NoCodeFoundError: No code found in the response
```

Does not take into account the previous question!

```
[ ]: data.groupby('data')['n_accessi_situazione_cliente'].mean()
```

```
[ ]: data
     2023-04-01     11.345411
     2023-04-02      0.000000
     2023-04-03    116.258376
     2023-04-04    127.270562
     2023-04-05    119.739106
     2023-04-06    113.296149
     2023-04-07     70.404305
     2023-04-08      7.395442
     2023-04-09      0.000000
     2023-04-10      0.066265
     2023-04-11    125.764773
     2023-04-12    124.607487
     2023-04-13    119.153977
     2023-04-14    112.646224
     2023-04-15     11.241158
     2023-04-16      0.150327
     2023-04-17    129.924915
     2023-04-18    119.468182
     2023-04-19    116.689812
     2023-04-20    114.164960
     2023-04-21    109.637813
     2023-04-22     10.618491
     2023-04-23      0.146341
     2023-04-24     78.819099
     2023-04-25      0.173387
     2023-04-26    132.395336
     2023-04-27    125.335616
     2023-04-28    125.087900
     2023-04-29     12.052993
     2023-04-30      0.000000
     Name: n_accessi_situazione_cliente, dtype: float64
```

Lazy? Italian works as well!

```
[ ]: pandas_ai(
         data,
         "Qual è la cod_age con il più alto valore di accesso a situazione cliente?␣
      ↪In che data?"
     )
```

```
[ ]: 'Il codice agenzia con il maggior numero di accessi alla situazione del cliente
     è il 39331 e la data in cui si è registrato il maggior numero di accessi per
     questo codice agenzia è il 4 aprile 2023.'
```

[ ]:
```
pandas_ai(
    data,
    "Qual è la cod_age con il più alto valore di accesso a situazione cliente e␣
 ↪Leonardo? In che date rispettivamente?"
)
```

[ ]:
```
'Il codice agenzia con il maggior numero di accessi alla situazione cliente e
Leonardo è il 39353. Le date in cui si sono verificati questi accessi sono:
inizio il 14 marzo 2023 e fine il 1 gennaio 1970.'
```

[ ]:
```
pandas_ai(
    data,
    "Qual è la cod_age con il più alto valore di accesso ai due portali␣
 ↪monitorati? In che date rispettivamente?"
)
```

[ ]:
```
'La cod_age con il valore di accesso più alto ai due portali monitorati è la
39353. Questo valore è stato registrato in diverse date, tra cui il 14 marzo
2023 e il 1 gennaio 1970.'
```