Acessibilidade e Inclusão na UFCG: Inteligência Artificial aplicada às dimensões educacional e física

Descrição: Projeto apoiado pelo Parque Tecnológica da Paraíba que tem, por objetivo, realizar a análise de dados das pessoas com deficiência matriculados na Universidade Federal de Campina Grande (UFCG) em que visa observar os padrões de notas, evasão e cursos utilizando Inteligência Artificial.

Metodologia:

O trabalho aqui apresentado seguirá as seguintes etapas metodológicas para atingir os objetivos almejados: a) Limpeza de dados; b) Selecionar os dados necessários para a pesquisa; c) Criação das variáveis analisadas; d) Criação de gráficos e d) Apresentação da conclusão. Cada uma dessas etapas - menos as etapas a) e b) - estarão presentes em cada situação analisada.

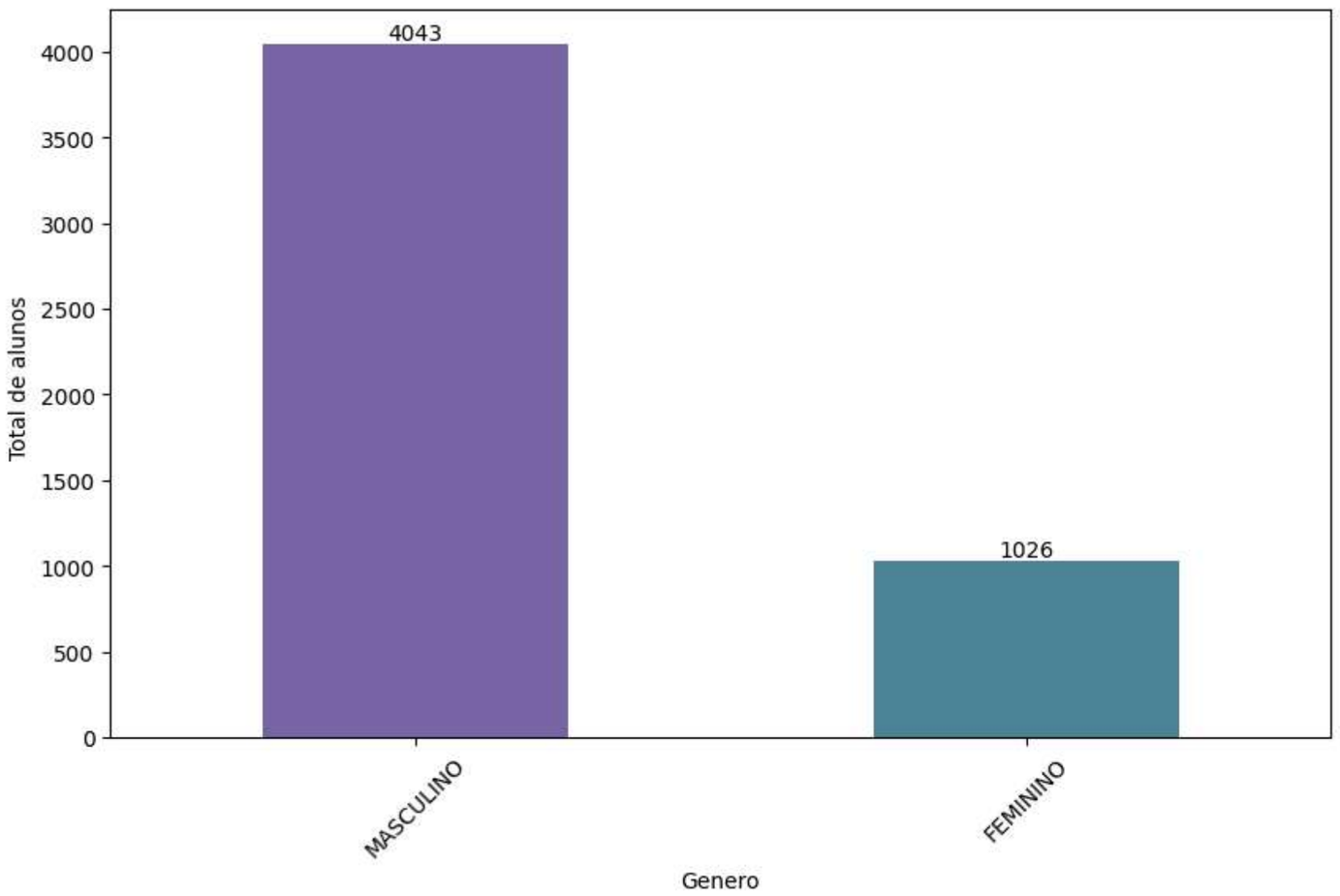
1. Análises necessárias para a pesquisa

a) Distribuição dos ingressantes por gênero
b) Distribuição dos ingressantes por raça
c) Distribuição dos ingressantes por cota
d) Distribuição geral dos ingressantes
e) Distribuição geral dos dados dos estudantes PCD
f) Distribuição das evasões
g) Distribuição dos estudantes graduados e não graduados

1.1. Análise (a)

```
ax = dados_analise['genero'].value_counts().plot.bar(

                                                color=['#7968A5', '#4e8397'],
                                                figsize=(10, 6))
plt.xticks(rotation=45)
ax.set_xlabel('Genero')
ax.set_ylabel('Total de alunos ')
for container in ax.containers:
  ax.bar_label(container, fmt='%.0f')
```




```
print(dados_analise['periodo_de_ingresso'].unique())
```

```
[2011.2 2006.1 1991.1 2023.1 1999.2 2012.2 2015.1 2013.2 1997.1 2024.2
 2016.1 2017.2 2014.2 1990.1 2020.1 2016.2 1996.2 2013.1 2006.2 2002.2
 2024.1 1984.1 1994.2 1999.1 2017.1 2010.1 2004.2 2009.1 2009.2 1993.1
 2019.2 2018.2 2014.1 2022.2 2007.2 2023.2 2012.1 2015.2 2001.2 2008.1
 1979.2 2003.1 1996.1 2005.2 2003.2 1988.1 1995.2 2020.2 1992.1 2005.1
 2011.1 2000.1 2010.2 1998.1 2021.1 2002.1 1989.1 2019.1 1994.1 2022.1
 1987.1 1986.1 1978.2 1980.1 2021.2 2018.1 1997.2 1993.2 2000.2 1995.1
 2004.1 1985.1 2007.1 2008.2 1977.2 1981.1 2001.1 1998.2 1983.1 1977.1
 1978.1 1982.1 1979.1]
```

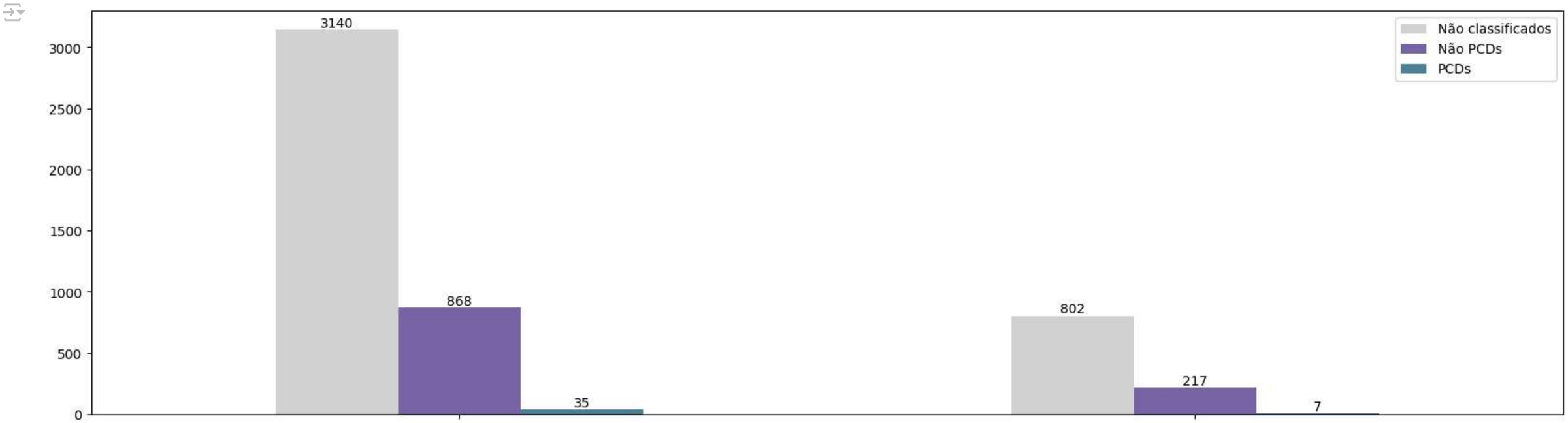
No gráfico 1, observamos uma grande maioria de homens que ingressam no curso de computação, analisando entre os dados de (1978 - 2024)

```
df = pd.DataFrame({
    'Não classificados': [totalHomensNI, totalMulheresNI],
    'Não PCDs': [totalHomens, totalMulheres],
    'PCDs': [totalHomensPCDs, totalMulheresPCDs],
}, index=['Homens', 'Mulheres'])

ax = df.plot.bar(
```

```
ax = df.plot.bar(

            color=['lightgray','#7968A5', '#4e8397'],
            figsize = (20,5.5),
            )
plt.xticks(rotation=45)
for container in ax.containers:
    ax.bar_label(container, fmt='%.0f')
```

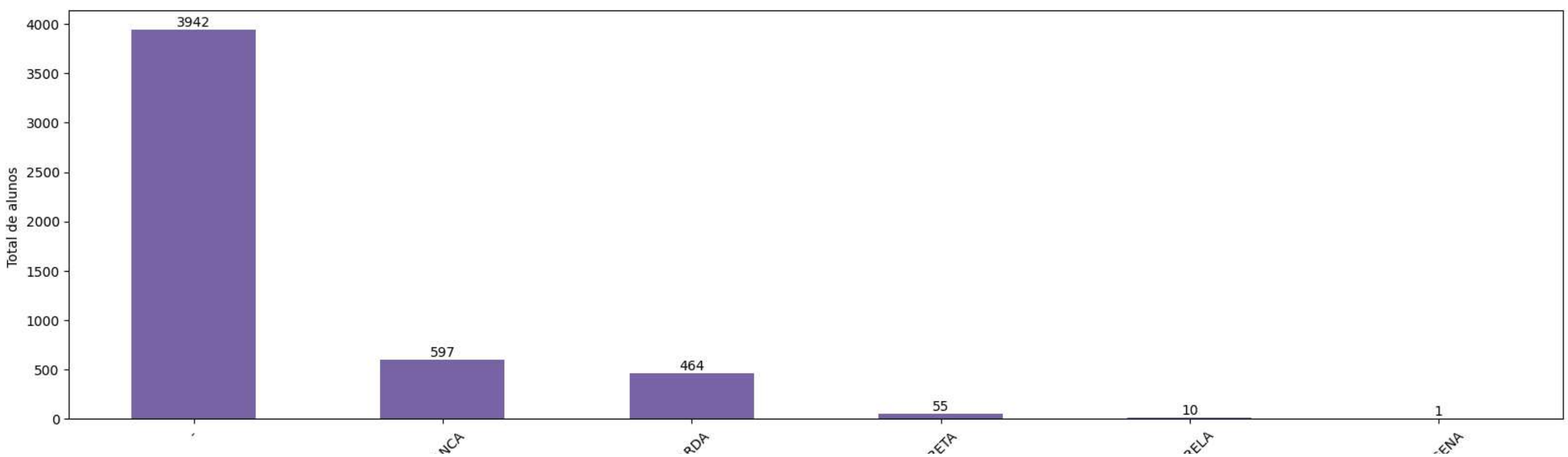


No gráfico 2, é possível observar a comparação dos estudantes que são PCD com os qque não são, dentro o intervalo de anos analisado, o total de ingressos de Homens PCDs foi muito baixo em comparação com os não PCD, além disso, a porcentagem de mulheres - que já era baixa - quando comparamos com mulheres PCDs, temos um valor ainda menor.

1.2. Análise (b)

```
ax = dados_analise['prac_cor'].value_counts().plot.bar(color=['#7968A5'],

                                                       figsize=(20,5.5))
plt.xticks(rotation=45)
ax.set_xlabel('Genero')
ax.set_ylabel('Total de alunos ')
for container in ax.containers:
  ax.bar_label(container, fmt='%.0f')
```

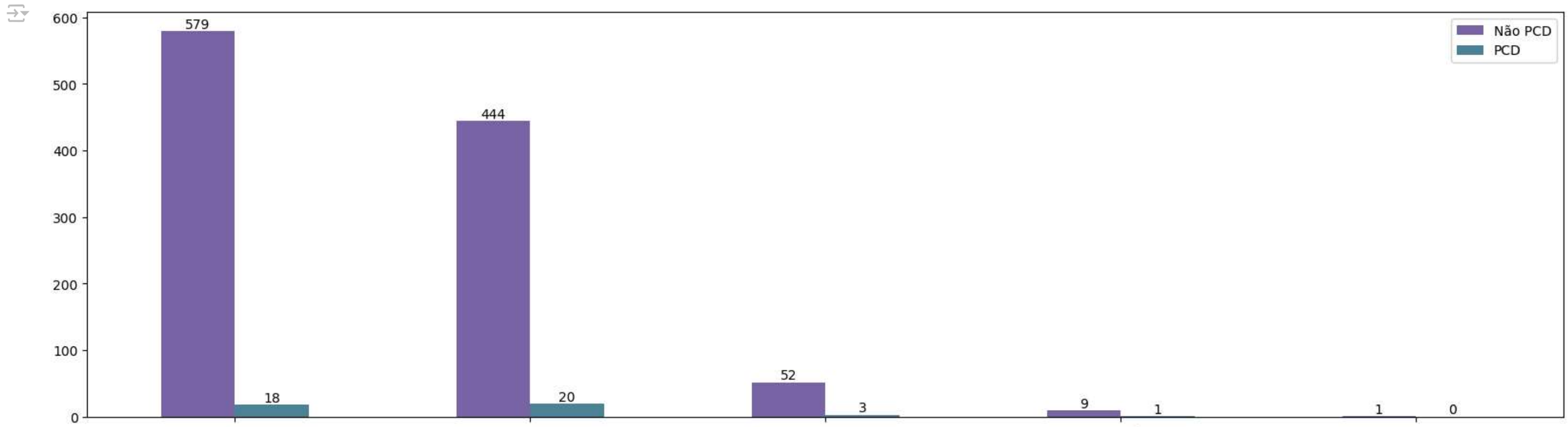


Os estudantes classificados com (-) não possuem auto-declaração de cor, assim, para a UFCG, temos uma maioria de estudantes classificados como brancos, seguidos de pardos, pretos, amarelos e indígenas.

```
df = pd.DataFrame({
    'Não PCD': [totalBrancos, totalPardos,  totalPretos, totalAmarelos, totalIndigenas],
    'PCD': [totalBrancosPCD,totalPardosPCD,  totalPretosPCD, totalAmarelosPCD, totalIndigenasPCD]
}, index=['Brancos', 'Pardos', 'Pretos', 'Amarelos', 'Indigenas'])

ax = df.plot.bar(

             color=['#7968A5', '#4e8397'],
             figsize = (20,5.5),
             )
plt.xticks(rotation=45)
for container in ax.containers:
    ax.bar_label(container, fmt='%.0f')
```

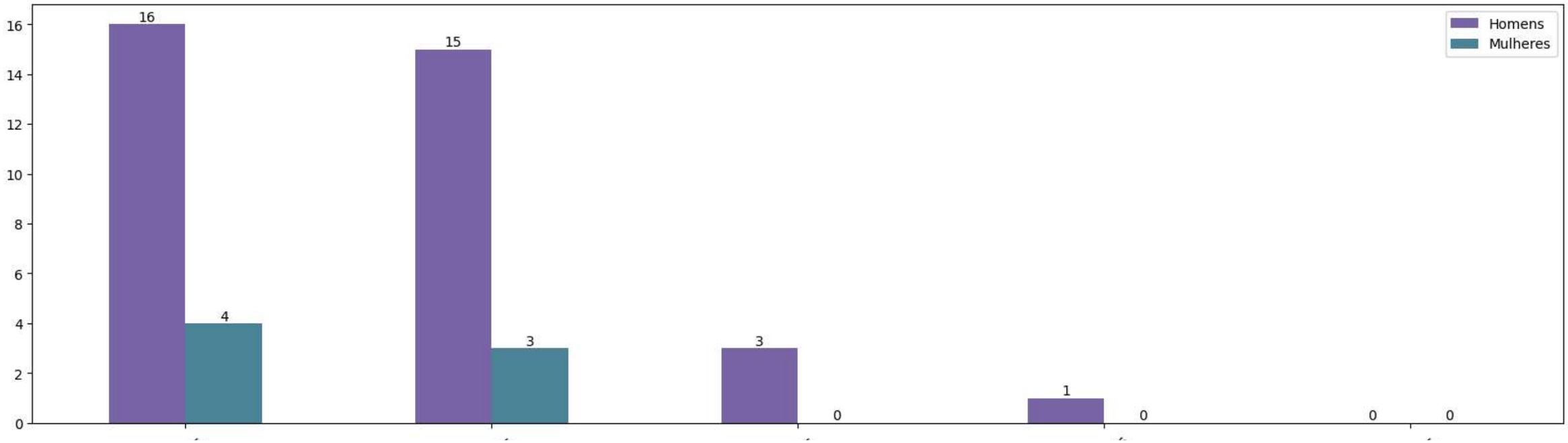


Para a classificação geral, observamos uma maior quantidade de alunos brancos que foram matriculados no curso, mas quando analisamos os alunos com deficiência, a maioria da classificação fica para os alunos pardos.

```
df = pd.DataFrame({
    'Homens': [totalPardosH, totalBrancosH, totalPretosH, totalAmarelosH, totalIndigenasH],
    'Mulheres': [totalPardosM, totalBrancosM, totalPretosM, totalAmarelosM, totalIndigenasM]
}, index=['Pardos', 'Brancos', 'Pretos', 'Amarelos', 'Indigenas'])

ax = df.plot.bar(

             color=['#7968A5', '#4e8397'],
             figsize = (20,5.5),
             )
plt.xticks(rotation=45)
for container in ax.containers:
    ax.bar_label(container, fmt='%.0f')
```



Quando realizado o recorte de gênero, observamos que, ainda, a maioria no curso é premoninantemente de homens, mesmo entre os estudantes PCDs.

1.3. Análise (c)

Tipos de cotas

a) Classificação geral

L1:

Estudantes de escola pública, com renda familiar per capita menor ou igual que um salário mínimo

L2:

Estudantes de escola pública, com renda familiar per capita menor ou igual que um salário mínimo e que se autodeclaram pretos, pardos ou indígenas.

L5:

Estudantes de escola pública, independente da renda.

L6:

Estudantes de escola pública, independente da renda e que se autodeclaram pretos, pardos ou indígenas.

L9:

Estudantes com deficiência, com renda familiar per capita menor ou igual que um salário mínimo e que tenham cursado o ensino médio em escola pública.

L10:

Estudantes com deficiência, que se autodeclaram pretos, pardos ou indígenas, com renda familiar per capita menor ou igual que um salário mínimo e que tenham cursado o ensino médio em escola pública.

L13:

Estudantes com deficiência, que tenham cursado o ensino médio em escola pública, independente da renda.

L14:

Estudantes com deficiência, que se autodeclaram pretos, pardos ou indígenas, que tenham cursado o ensino médio em escola pública, independente da renda.

b) Classificação dada pela UFCG

LI_PPI

Candidatos autodeclarados pretos, pardos ou indígenas, independentemente da renda, que tenham cursado integralmente o ensino médio em escolas públicas

LB_PPI

Candidatos autodeclarados pretos, pardos ou indígenas, com renda familiar bruta per capita igual ou inferior a 1 salário mínimo e que tenham cursado integralmente o ensino médio em escolas públicas

LB_EP

Candidatos com renda familiar bruta per capita igual ou inferior a 1 salário mínimo que tenham cursado integralmente o ensino médio em escolas públicas

LI_EP

Candidatos que, independentemente da renda, tenham cursado integralmente o ensino médio em escolas públicas

LI_Q

Candidatos autodeclarados quilombolas, independentemente da renda, tenham cursado integralmente o ensino médio em escolas públicas

LB_PCD

Candidatos com deficiência, que tenham renda familiar bruta per capita igual ou inferior a 1 salário mínimo e que tenham cursado integralmente o ensino médio em escolas públicas

LI_PCD

Candidatos autodeclarados pretos, pardos ou indígenas, independentemente da renda, que tenham cursado integralmente o ensino médio em escolas públicas

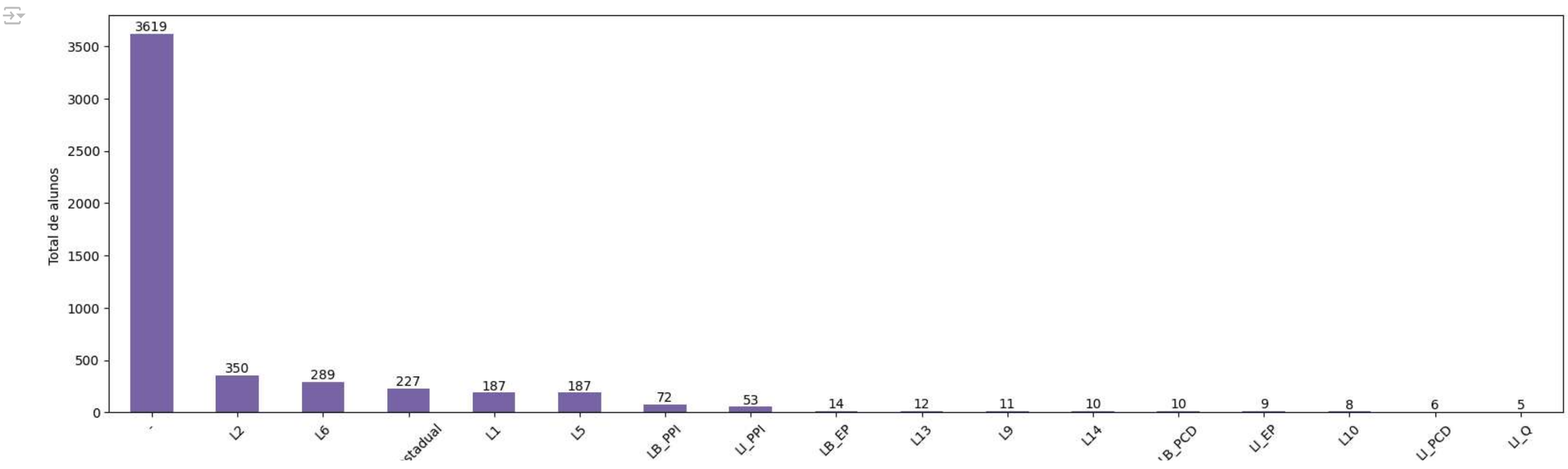
Bônus estadual:

Estudantes paraibanos podem ter um acréscimo de até 10% na nota de classificação.

Fonte: Link

```
ax = dados_analise['politica_afirmativa'].value_counts().plot.bar(color=['#7968A5'],

                                                  figsize=(20,5.5))
plt.xticks(rotation=45)
ax.set_xlabel('Genero')
ax.set_ylabel('Total de alunos ')
for container in ax.containers:
  ax.bar_label(container, fmt='%.0f')
```

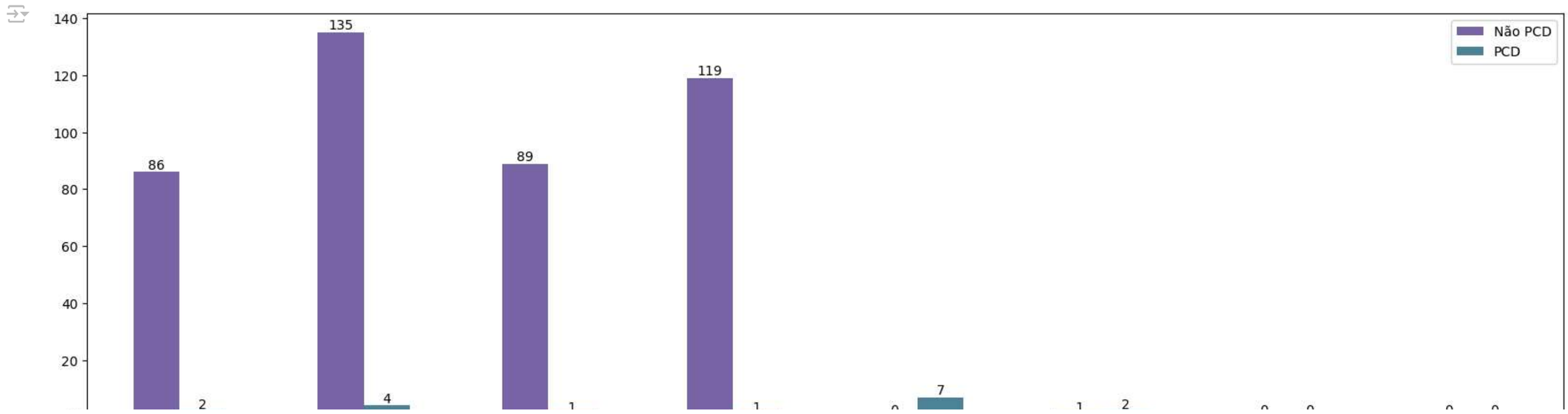


Nessa classificação, é possível observar que a maioria dos ingressos se deu por meio da ampla concorrência - candidatos sem cotas - Além disso, também existe uma presença maior de candidatos vindos de escolas públicas, com renda inferior ou igual a um salário mínimo e autodeclarados pretos, pardos ou indígenas.

```
df = pd.DataFrame({
    'Não PCD': [politica_afirmativa_l1, politica_afirmativa_l2, politica_afirmativa_l5, politica_afirmativa_l6, politica_afirmativa_l9, politica_afirmativa_l10,
    'PCD': [politica_afirmativa_l1_pcd, politica_afirmativa_l2_pcd, politica_afirmativa_l5_pcd, politica_afirmativa_l6_pcd, politica_afirmativa_l9_pcd, politica_
}, index=['L1', 'L2', 'L5', 'L6', 'L9', 'L10', 'L13', 'L14', ])

ax = df.plot.bar(
            color=['#7968A5', '#4e8397'],
```

```
            figsize = (20,5.5),
            )
plt.xticks(rotation=45)
for container in ax.containers:
    ax.bar_label(container, fmt='%.0f')
```

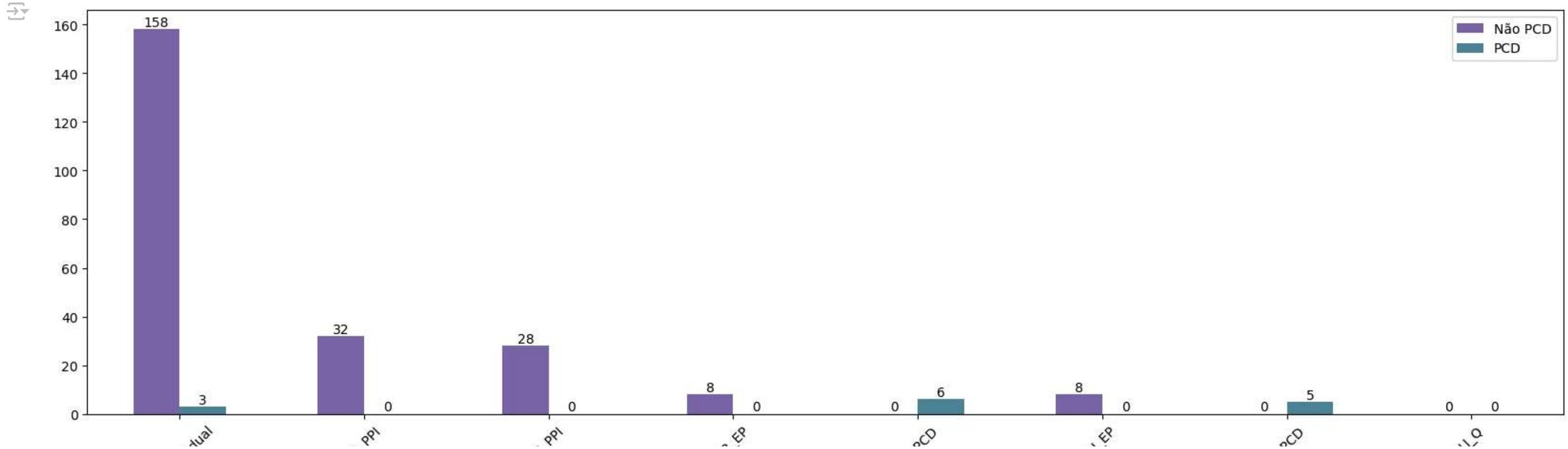


Quando observado os alunos apenas que entraram por cota (PCDs e não PCDs), temos uma entrada maior dos alunos não PCDs ingressando no curso por meio da política afirmativa, ainda se mantendo o padrão da cota L2 possuir o maior número de alunos ingressantes.

```
df = pd.DataFrame({
    'Não PCD': [politica_afirmativa_bonus, politica_afirmativa_lbppi, politica_afirmativa_lippi, politica_afirmativa_lbep, politica_afirmativa_lbpcd, politica_afirmativa_liep, politica_afirmat
    'PCD': [politica_afirmativa_bonus_pcd, politica_afirmativa_lbppi_pcd, politica_afirmativa_lippi_pcd, politica_afirmativa_lbep_pcd, politica_afirmativa_lbpcd_pcd, politica_afirmativa_liep_p
}, index=['Bon. Estadual', 'LB_PPI', 'LI_PPI', 'LB_EP', 'LB_PCD', 'LI_EP', 'LI_PCD', 'LI_Q'])

ax = df.plot.bar(

            color=['#7968A5', '#4e8397'],
            figsize = (20,5.5),
            )
plt.xticks(rotation=45)
for container in ax.containers:
    ax.bar_label(container, fmt='%.0f')
```



Para a classificação disponibilizada pela UFCG, temos um número expressivo de alunos entrando com o bônus estadual, mas um número bem menor de alunos PCDs presentes nessas categorias.

1.4. Análise (d)

```
df_tiposDeDeficiencia.columns = ["Código da Deficiência", "Deficiência"]

nw_tiposDeDeficiencia = df_tiposDeDeficiencia.style.set_table_styles([
    {'selector' : 'thead','props' : [('background-color','#4e8397')]},
    {'selector' : 'tbody tr:hover','props' : [('background-color','#97afb9'), ('color', 'white')]},
    {'selector' : 'caption', 'props': [{'font-size', '1.5em'}]}
  ])
nw_tiposDeDeficiencia
```

	Código da Deficiência	Deficiência
0	A10	Deficiência auditiva
1	A11	Surdez
2	A21	Cegueira
3	A22	Baixa visão ou visão monocular
4	A30	Deficiência física
5	A41	Surdocegueira
6	A50	Deficiência intelectual
7	B10	Transtorno do espectro autista
8	B11	Síndrome de Asperger
9	B12	Síndrome de Rett
10	B13	Transtorno Desintegrativo da Infância

```
df_dadosEstudantesDeficiencias.rename(columns={'deficiencias': 'Código da Deficiência'}, inplace=True)
df_dadosEstudantesDeficiencias['Código da Deficiência'] = df_dadosEstudantesDeficiencias['Código da Deficiência'].apply(lambda x: ', '.join(x) if isinstance(x, l

relation_dadosEstudantes = pd.merge(df_dadosEstudantesDeficiencias, df_tiposDeDeficiencia, how = 'inner', on = 'Código da Deficiência')
df_total = relation_dadosEstudantes.groupby('Deficiência').size().reset_index(name='Total Estudantes')
```

```
nw_totalEstudantesComDeficiencia = df_total.style.set_table_styles([
    {'selector' : 'thead','props' : [('background-color','#4e8397')]},
    {'selector' : 'tbody tr:hover','props' : [('background-color','#97afb9'), ('color', 'white')]},
    {'selector' : 'caption', 'props': [{'font-size', '1.5em'}]}
  ])

nw_totalEstudantesComDeficiencia
```

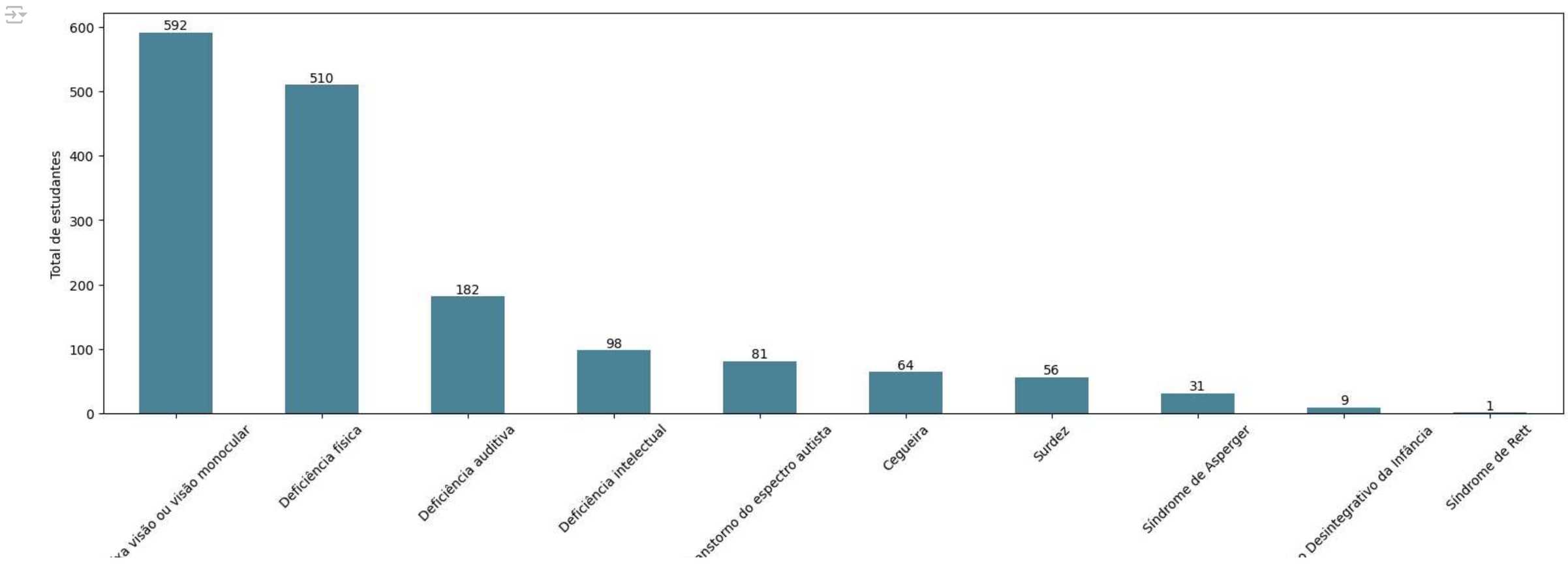
	Deficiência	Total Estudantes
0	Baixa visão ou visão monocular	592
1	Cegueira	64
2	Deficiência auditiva	182
3	Deficiência física	510
4	Deficiência intelectual	98
5	Surdez	56
6	Síndrome de Asperger	31
7	Síndrome de Rett	1
8	Transtorno Desintegrativo da Infância	9
9	Transtorno do espectro autista	81

Para os campis da UFCG, é possível observar a predominância de alunos com baixa visão, seguido de cegueira e deficiência auditiva.

```
ax = relation_dadosEstudantes["Deficiência"].value_counts().plot.bar(color='#4e8397',
                                                                     figsize=(20,5.5))

ax.set_xlabel('Descrição da deficiência')
ax.set_ylabel('Total de estudantes')
plt.xticks(rotation=45)

for container in ax.containers:
  ax.bar_label(container, fmt='%.0f')
```



1.6. Análise (f)

```
df = pd.DataFrame({
    'Não PCD': [totalGraduados, totalRegular, totalAbandono, total3Reprov, totalNovoIngressoMesmo, totalCancelamentoMat, totalReprovFaltas,
                totalNaoCompareceuCadastro, totalCancelamentoSolic, totalNovoIngressoOutro, totalTransferido, totalRemanej , totalCancelamentoMudanca,
                totalNaoFezMat, totalAguardando, totalConcluido, totalCancelamentoJudicial, totalConvenio, totalNaoCompareceuRemanej],
    'PCD': [totalGraduadosPCD, totalRegularPCD, totalAbandonoPCD, total3ReprovPCD, totalNovoIngressoMesmoPCD,totalCancelamentoMatPCD, totalReprovFaltasPCD,
            totalNaoCompareceuCadastroPCD, totalCancelamentoSolicPCD, totalNovoIngressoOutroPCD, totalTransferidoPCD, totalRemanejPCD, totalCancelamentoMudancaP(
            totalNaoFezMatPCD,  totalAguardandoPCD, totalConcluidoPCD, totalCancelamentoJudicialPCD, totalConvenioPCD, totalNaoCompareceuRemanejPCD],
```

```
}, index=['GRADUADO', 'REGULAR', 'CANCELAMENTO POR ABANDONO', 'CANCELADO 3 REPROV MESMA DISCIPLINA', 'CANCELADO NOVO INGRESSO MESMO CURSO', 'CANCELAMENTO DE MATI
          'NAO COMPARECEU CADASTRO', 'CANCELAMENTO P/ SOLICITACAO ALUNO', 'CANCELADO NOVO INGRESSO OUTRO CURSO', 'TRANSFERIDO PARA OUTRA IES', 'REMANEJADO CURSO
           'INGRESSANTE NAO FEZ 1Âª MATRICULA','AGUARDANDO CADASTRAMENTO', 'CONCLUIDO - NAO COLOU GRAU', 'CANCELAMENTO P/ DECISAO JUDICIAL', 'CUMPRIMENTO CONVEN


ax = df.plot.bar(

             color=['#7968A5', '#4e8397'],
             figsize = (14,4),
             )
plt.xticks(rotation=45)
for container in ax.containers:
    ax.bar_label(container, fmt='%.0f')
```