In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
from matplotlib import pyplot as plt
import pylab
import sys
import warnings
import statistics
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
```

In [2]:
```
# Comandos para suprimir os warnings
if not sys.warnoptions:
    warnings.simplefilter("ignore")
```

In [4]:
```
# Leando o arquivo

df1 = pd.read_csv('cars_train.csv', encoding='UTF-16', sep='\t')
```

In [5]:
```
df1.head(3)
```

Out[5]:

	id	num_fotos	marca	modelo	versao
0	300716223898539419613863097469899222392	8.0	NISSAN	KICKS	1.6 16V FLEXSTART SL 4P XTRONIC
1	279639842134129588306469566150288644214	8.0	JEEP	COMPASS	2.0 16V FLEX LIMITED AUTOMÁTICO
2	564144608106210489002956782365381719 81	16.0	KIA	SORENTO	2.4 16V GASOLINA EX 7L AWD AUTOMÁTICO

3 rows × 29 columns

In [6]:
```
# Explorando o arquivo parte 1
type(df1)
```

Out[6]:
```
pandas.core.frame.DataFrame
```

In [ ]:
```
df1.info()
```

In [7]:
```
# Altarando o tipo de dados
df1['ano_modelo'] = df1['ano_modelo'].astype(np.int16)
df1['ano_de_fabricacao'] = df1['ano_de_fabricacao'].astype(np.int16)
df1['num_portas'] = df1['num_portas'].astype(np.int16)
df1['ano_modelo'] = df1['ano_modelo'].astype(np.int16)
df1['preco'] = df1['preco'].astype(np.int32)
df1['entrega_delivery'] = df1['entrega_delivery'].astype(np.int32)
df1['troca'] = df1['troca'].astype(np.int32)
df1['elegivel_revisao'] = df1['elegivel_revisao'].astype(np.int32)
```

In [ ]:
```
# Explorando o arquivo parte 2: Possui valores nulos?  muitos valores nulos

# Das 29.584 linhas temos: 6 colunas que serão excluida por falta de dados:
# veiculo_único_dono + revisoes_concessionaria + ipva_pago + veiculo_licenc
# garantia_de_fábrica + revisoes_dentro_agenda + veiculo_alienado + troca
# excluiremos as linhas da coluna fotos que estão com valores nulos

df1.isnull().sum()
```

In [8]:
```
df2 = df1.drop( columns=['id','veiculo_único_dono','revisoes_concessionaria
df2 = df2.dropna(how='any', axis=0)
```

In [ ]:
```
# Explorando o arquivo parte 3: Possui valores na?
df2.isnull().sum()
```

In [ ]:
```
df2.info()
```

In [ ]:
```
df2.describe()
```

In [ ]:
```
df2.agg({"preco": ["min", "mean", "max", "std", "count"]}).astype(np.int32)
```

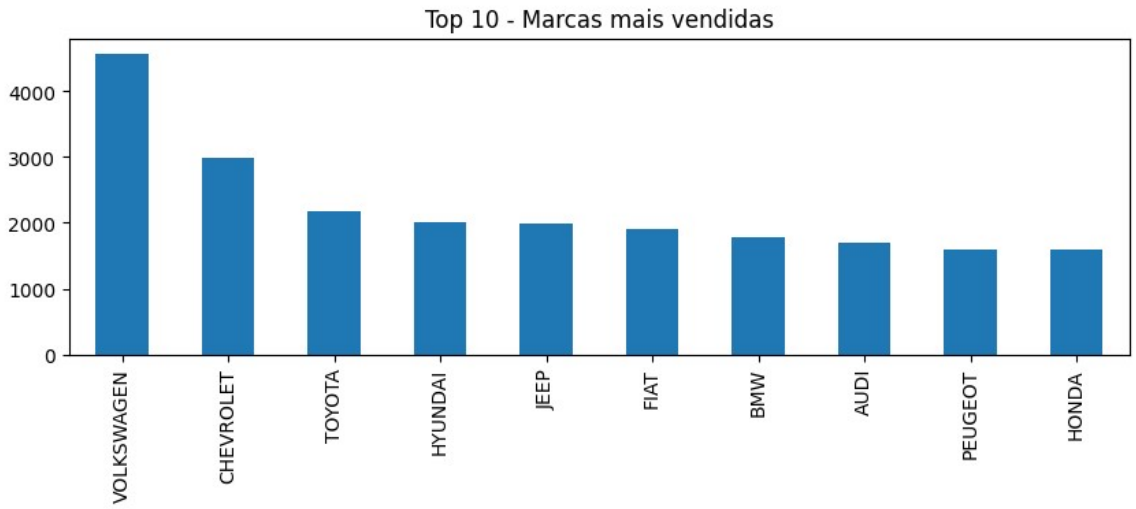
In [ ]:
```
# Análise de correlação - Mapa de calor
#  Dada a alta correção e provavel efeito de colinearidade precisamos elimi
#  optamos pela coluna Ano de Fabricação.

matriz_correlacao = df1.corr( )
plt.figure(figsize = (10, 7))
sns.heatmap(matriz_correlacao, annot = True, fmt='.1f')
```

In [9]:
```
df3 = df2.drop( columns=['ano_de_fabricacao'])
```

In [ ]:
```
df3.head(3)
```

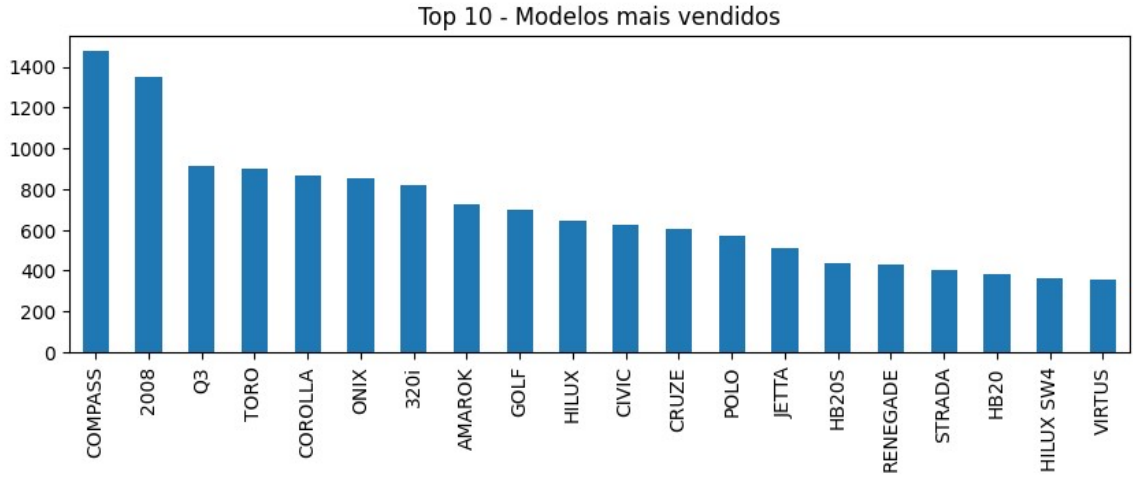
In [10]:
```
# Quais São as marcas mais vendidas? - A partir dessa perguntas descobrimos
# de automóveis deve priorizar negociações com as 5 primeiras marcas, pois
# os veículos possuem menor tempo de permanência em estoque.

df3_Marcas_mais_vendidas = df3['marca'].value_counts()[:10]
#print(df3_Marcas_mais_vendidas)
df3_Marcas_mais_vendidas.plot(kind='bar', figsize=(10, 3))
plt.title('Top 10 - Marcas mais vendidas')
```

Out[10]: Text(0.5, 1.0, 'Top 10 - Marcas mais vendidas')



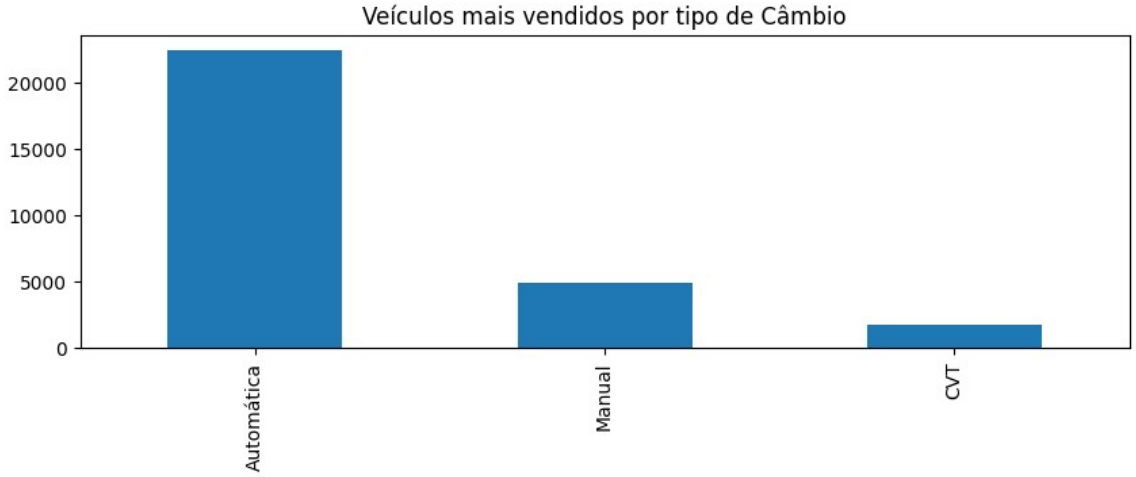
In [34]:
```
# Quais São os 10 modelos de carros mais vendidos? - A partir dessa pergunt
# de automóveis deve priorizar negociações com as 15 primeiras marcas, poi
# os veículos possuem menor tempo de permanência em estoque. Perceba que os
# ocupam as primeiras posições no top 5 modelos mais vendidos, contido ao l
# mais vendidos temos mais modelos volkswagem dentras as demais marcas.


df3_Modelos_mais_vendidos = df3['modelo'].value_counts()[:20]
#print(df3_Modelos_mais_vendidos)
df3_Modelos_mais_vendidos.plot(kind='bar', figsize=(10, 3))
plt.title('Top 10 - Modelos mais vendidos')
```

Out[34]: Text(0.5, 1.0, 'Top 10 - Modelos mais vendidos')



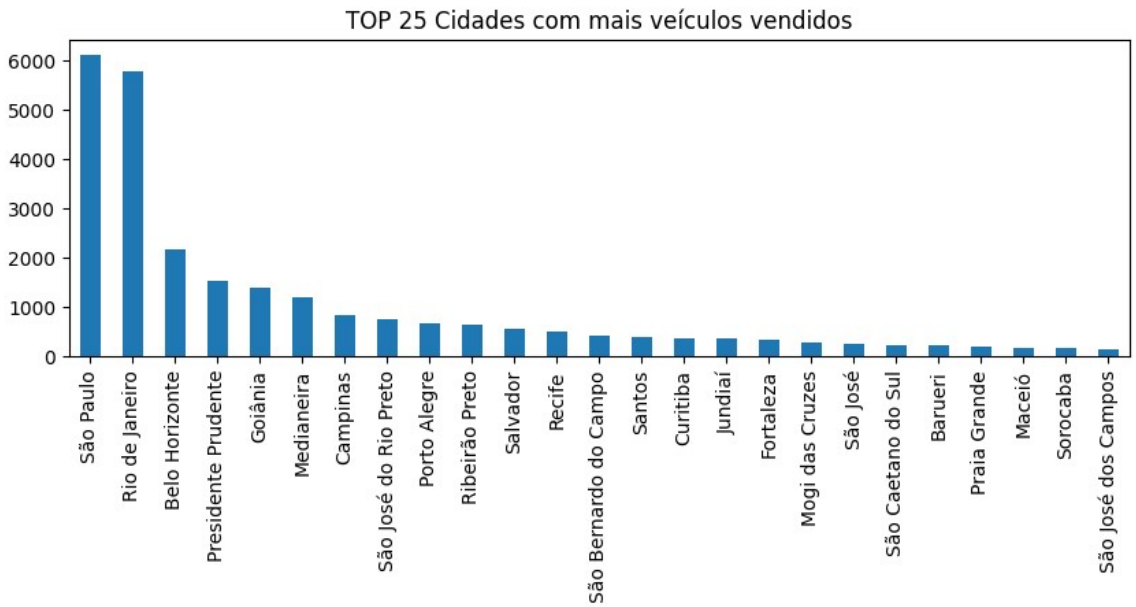
In [35]:
```
#Por esse gráfico é possível inferir que os veículos com câmbio automáticos
# da negociação. Logo deve-se dar preferência para os veículos automáticos
df3_veiculos_Cambio = df3['cambio'].value_counts()[:3]
#print(df3_veiculos_Cambio)
df3_veiculos_Cambio.plot(kind='bar', figsize=(10, 3))
plt.title('Veículos mais vendidos por tipo de Câmbio')
```

Out[35]: Text(0.5, 1.0, 'Veículos mais vendidos por tipo de Câmbio')



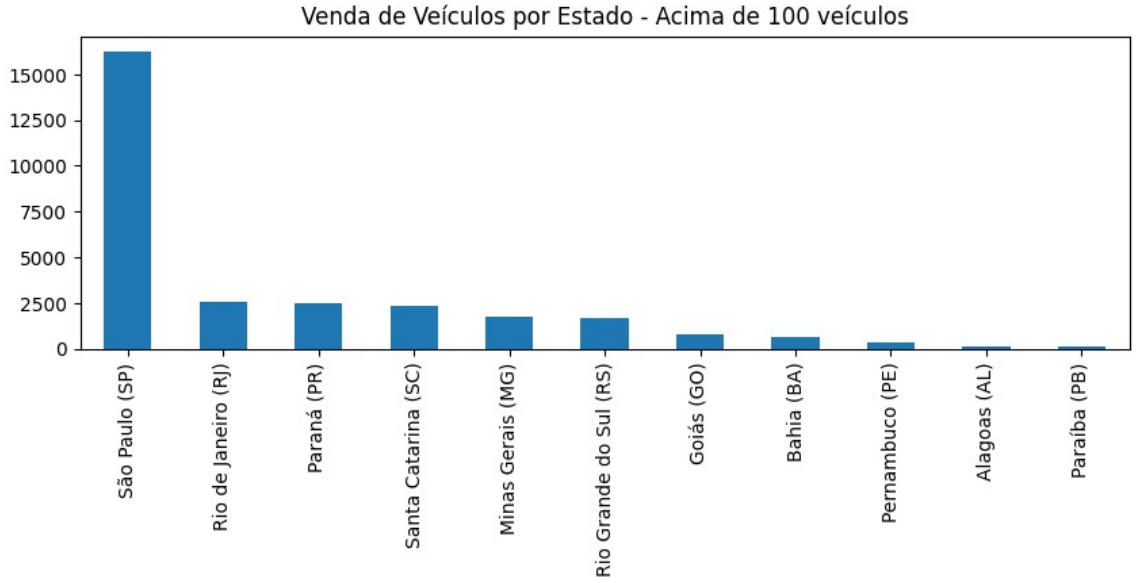
In [36]:
```
# 25 melhores cidades para vender veículos.

def3_cidade_mais_vendas = df3['cidade_vendedor'].value_counts()[:25]
#print(def3_cidade_mais_vendas)
def3_cidade_mais_vendas.plot(kind='bar', figsize=(10, 3))
plt.title('TOP 25 Cidades com mais veículos vendidos')
```

Out[36]: Text(0.5, 1.0, 'TOP 25 Cidades com mais veículos vendidos')



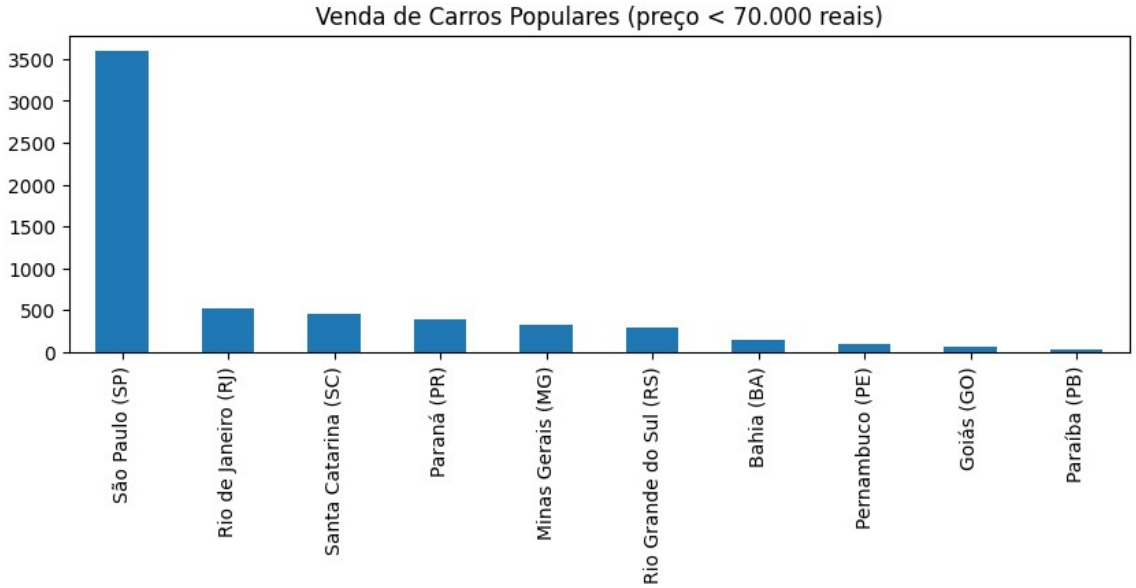
In [37]:
```
# Estados com quantidade de vendas superior a 100 veículos.

def3_estado_melhor_venda = df3['estado_vendedor'].value_counts()[:11]
#print(def3_estado_melhor_venda)
def3_estado_melhor_venda.plot(kind='bar', figsize=(10, 3))
plt.title('Venda de Veículos por Estado - Acima de 100 veículos')
```

Out[37]: Text(0.5, 1.0, 'Venda de Veículos por Estado - Acima de 100 veículos')



In [38]:
```
# Venda de Carros populares

#Qual o melhor estado cadastrado na base de dados para se vender um carro d
# O melhor estado para vender carros populares ainda é São Paulo por ser um
# A partir do terceiro Estado é possível perceber que a venda de carros pop
# relação direta com o tamanho da população.

df3_carros_populares = df3[df3['preco']<70000]['estado_vendedor'].value_cou
#print(df3_carros_populares)
df3_carros_populares.plot(kind='bar', figsize=(10, 3))
plt.title('Venda de Carros Populares (preço < 70.000 reais)')
```

Out[38]: Text(0.5, 1.0, 'Venda de Carros Populares (preço < 70.000 reais)')



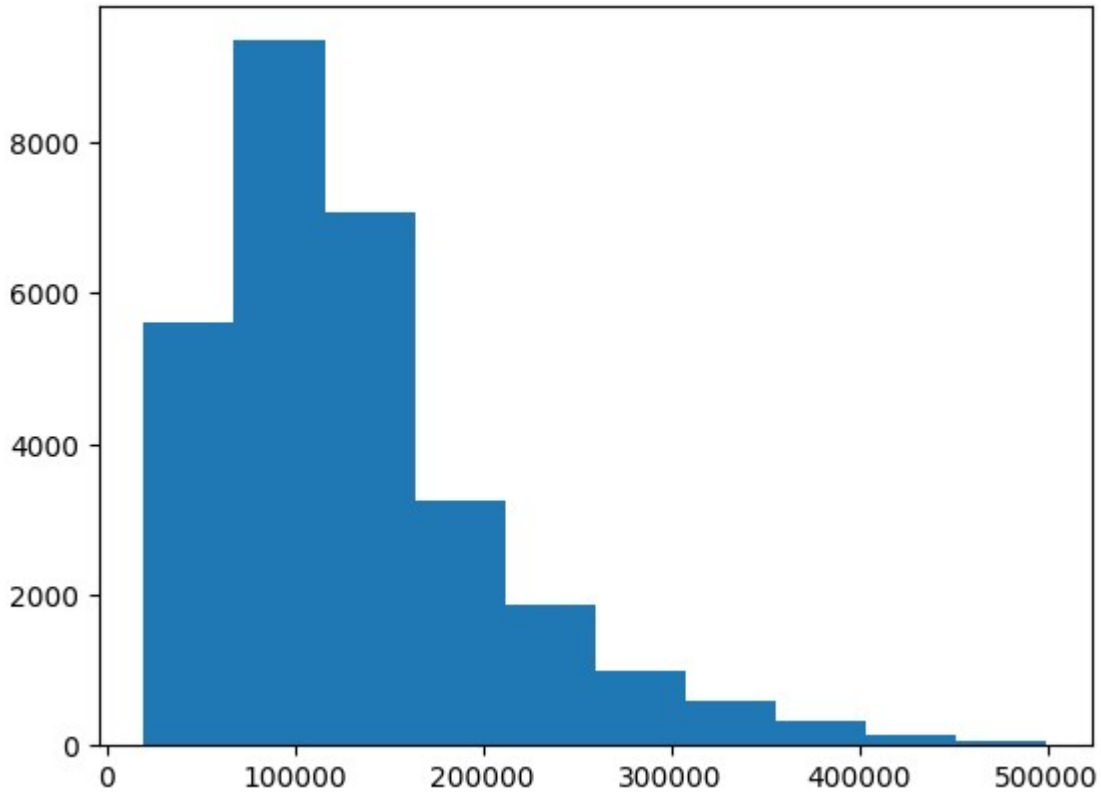
In [11]:
```
#Qual o melhor estado para se comprar uma picape com transmissão automática
# O melhor Estado para comprar uma picape com transmissão automática é São
# O segundo melhor Estado é o Paraná com 348 Picapes.

df3_Picape_automatica = df3.groupby((df3['tipo']=='Picape') & (df3['cambio'
print(df3_Picape_automatica)
```

```
       estado_vendedor
False  São Paulo (SP)              14563
       Rio de Janeiro (RJ)          2222
       Paraná (PR)                  2162
       Santa Catarina (SC)          2019
       Minas Gerais (MG)            1551
       Rio Grande do Sul (RS)       1448
       Goiás (GO)                    676
       Bahia (BA)                    535
       Pernambuco (PE)               304
       Alagoas (AL)                  110
       Paraíba (PB)                  104
       Rio Grande do Norte (RN)       89
       Ceará (CE)                     69
       Pará (PA)                      55
       Amazonas (AM)                  51
       Mato Grosso do Sul (MS)        30
       Mato Grosso (MT)               25
       Acre (AC)                      23
       Espírito Santo (ES)            21
       Sergipe (SE)                   19
       Tocantins (TO)                 17
       Maranhão (MA)                   7
       Rondônia (RO)                   4
       Roraima (RR)                    2
       Piauí (PI)                      1
True   São Paulo (SP)               1712
       Paraná (PR)                   348
       Rio de Janeiro (RJ)           318
       Santa Catarina (SC)           283
       Minas Gerais (MG)             211
       Rio Grande do Sul (RS)        198
       Goiás (GO)                    102
       Bahia (BA)                     68
       Pernambuco (PE)                14
       Alagoas (AL)                   12
       Acre (AC)                       6
       Mato Grosso (MT)                6
       Mato Grosso do Sul (MS)         5
       Sergipe (SE)                    5
       Paraíba (PB)                    4
       Piauí (PI)                      4
       Tocantins (TO)                  3
       Rio Grande do Norte (RN)        1
Name: estado_vendedor, dtype: int64
```

In [ ]:
```
# Tratando outliers
```

In [12]:
```
df4 = df3
```

In [39]:
```
plt.hist(df4['preco'])
plt.show()
```



In [13]:
```
#quantidade de linhas: 29407
df4['preco'].shape
```

Out[13]: (29407,)

In [14]:
```
#outliers Superiores
df4_out = df4[df4['preco']>500000].value_counts()
df4_out.shape
```

Out[14]: (89,)

In [15]:
```
#outliers Superiores
df4_out = df4[df4['preco']<20000].value_counts()
df4_out.shape
```

Out[15]: (28,)

In [16]:
```
# Removendo outliers
df4 = df4.loc[(df4['preco'] > 20000) & (df4['preco'] < 500000)]
```

In [17]:
```
df4.shape
```

Out[17]: (29290, 19)

In [ ]:
```
plt.hist(df4['preco'])
plt.show()
```

In [25]:
```
# Transformando variáveis categóricas em numéricas:

categorical_features = ['marca','modelo', 'versao', 'cambio', 'tipo', 'blin
```

In [26]:
```
df5 = df4.drop(columns=["num_fotos","cidade_vendedor","estado_vendedor"])
```

In [27]:
```
one_hot = OneHotEncoder()
transformer = ColumnTransformer([("one_hot",
                                  one_hot,
                                  categorical_features)],
                                  remainder='passthrough' )

transformed_x = transformer.fit_transform(df5)
transformed_x
```

Out[27]:
```
<23432x2324 sparse matrix of type '<class 'numpy.float64'>'
        with 374912 stored elements in Compressed Sparse Row format>
```

In [ ]:
```
np.random.seed(42)
X_train, X_test, Y_train, Y_test = train_test_split(transformed_x, y, test_
model.fit(X_train, Y_train)

model.score(X_test, Y_test)
```