This jupyter notebook is prepared by "Yuyang Zhang".

1. Run the block below to upload the dataset. (Note that the file list gets refreshed every time your runtime is disconnected. Simply run this when you return to upload the file again using the files API. Once you run, it should wait for you to upload the file. (1pt)

```
from google.colab import files
uploaded = files.upload()
```

选择文件 startup_info_.csv

- **startup_info_.csv**(text/csv) - 168764 bytes, last modified: 2023/1/27 - 100% done

```
Saving startup_info_.csv to startup_info_.csv
```

2. Import numpy, pandas, matplotlib.pyplot and seaborn packages. (2pt)

If you need additional packages, you can import it on the go in any code-block below.

```
#TODO
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import scipy.stats as st
from sklearn.preprocessing import LabelEncoder
from scipy.stats import skew
from sklearn.preprocessing import MaxAbsScaler
from sklearn.preprocessing import MinMaxScaler
```

3. Import the dataset into a pandas dataframe. Then report how many rows and columns are present in the dataset. (2pt)

```
#TODO
df = pd.read_csv('/content/startup_info_.csv')
df.shape
```

```
(923, 28)
```

4. Call the describe method to see summary statistics of the numerical attribute columns. (1pt)

```
#TODO
df.describe()
```

	Unnamed: 0	latitude	longitude	labels	age_first_funding_year	age_last_funding_year	age_first_milesto
count	923.000000	923.000000	923.000000	923.000000	923.000000	923.000000	77
mean	572.297941	38.517442	-103.539212	0.646804	2.235630	3.931456	
std	333.585431	3.741497	22.394167	0.478222	2.510449	2.967910	
min	1.000000	25.752358	-122.756956	0.000000	-9.046600	-9.046600	-1
25%	283.500000	37.388869	-122.198732	0.000000	0.576700	1.669850	
50%	577.000000	37.779281	-118.374037	1.000000	1.446600	3.528800	
75%	866.500000	40.730646	-77.214731	1.000000	3.575350	5.560250	
max	1153.000000	59.335232	18.057121	1.000000	21.895900	21.895900	2

5.1 List all attribute columns (1pt)

```
#TODO
df.columns
```

```
Index(['Unnamed: 0', 'state_code', 'latitude', 'longitude', 'zip_code', 'id',
       'city', 'Unnamed: 6', 'name', 'labels', 'founded_at', 'closed_at',
       'first_funding_at', 'last_funding_at', 'age_first_funding_year',
       'age_last_funding_year', 'age_first_milestone_year',
       'age_last_milestone_year', 'relationships', 'funding_rounds',
       'funding_total_usd', 'milestones', 'state_code.1', 'category_code',
       'object_id', 'avg_participants', 'is_top500', 'status'],
      dtype='object')
```

5.2 The "Unnamed: 0","Unnamed: 6", "state_code.1" and "object_id" feature columns are not useful. Drop them in-place. (1pt)

```
#TODO
df.drop(columns=['Unnamed: 0', 'Unnamed: 6', 'state_code.1', 'object_id'], inplace=True)
df.columns
```

```
Index(['state_code', 'latitude', 'longitude', 'zip_code', 'id', 'city', 'name',
       'labels', 'founded_at', 'closed_at', 'first_funding_at',
       'last_funding_at', 'age_first_funding_year', 'age_last_funding_year',
       'age_first_milestone_year', 'age_last_milestone_year', 'relationships',
       'funding_rounds', 'funding_total_usd', 'milestones', 'category_code',
       'avg_participants', 'is_top500', 'status'],
      dtype='object')
```

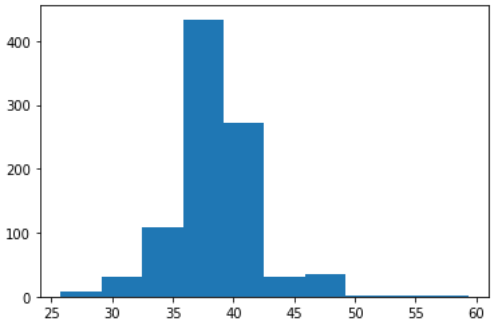
6.1 Show all the numeric columns and save it to a new dataframe. (2pt)

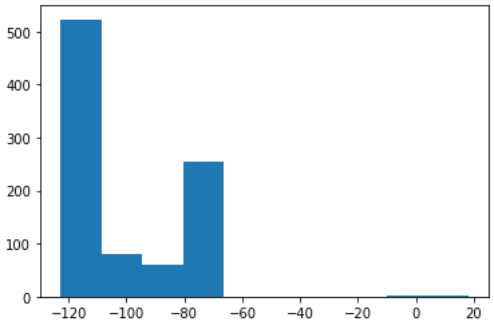
```
#TODO
numeric_features = df.select_dtypes(include=[np.number])
numeric_features.columns
```

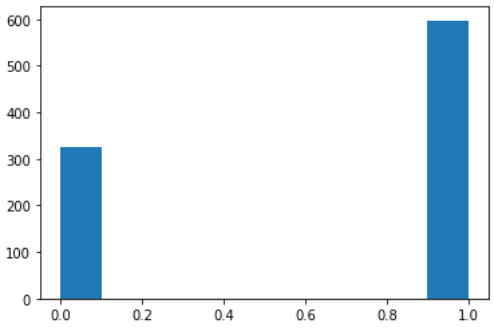
```
Index(['latitude', 'longitude', 'labels', 'age_first_funding_year',
       'age_last_funding_year', 'age_first_milestone_year',
       'age_last_milestone_year', 'relationships', 'funding_rounds',
       'funding_total_usd', 'milestones', 'avg_participants', 'is_top500'],
      dtype='object')
```

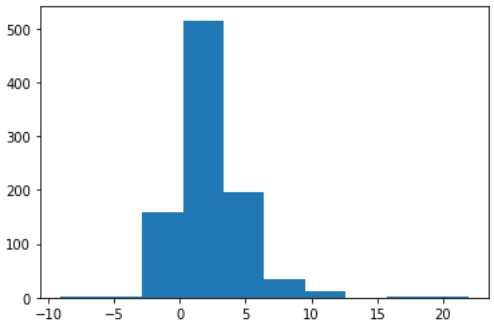
6.2 Plot distributions of the numeric columns using histogram and record the skew of each distribution. (Note: positive value = right skewed, negative value = left skewed) (4pt)

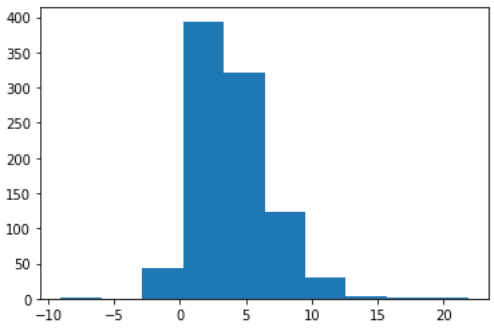
```
#TODO
plt.hist(numeric_features['latitude'])
plt.show()
plt.hist(numeric_features['longitude'])
plt.show()
plt.hist(numeric_features['labels'])
plt.show()
plt.hist(numeric_features['age_first_funding_year'])
plt.show()
plt.hist(numeric_features['age_last_funding_year'])
plt.show()
plt.hist(numeric_features['age_first_milestone_year'])
plt.show()
plt.hist(numeric_features['age_last_milestone_year'])
plt.show()
plt.hist(numeric_features['relationships'])
plt.show()
plt.hist(numeric_features['funding_rounds'])
plt.show()
plt.hist(numeric_features['funding_total_usd'])
plt.show()
plt.hist(numeric_features['milestones'])
plt.show()
plt.hist(numeric_features['avg_participants'])
plt.show()
plt.hist(numeric_features['is_top500'])
plt.show()
numeric_features.skew()
```

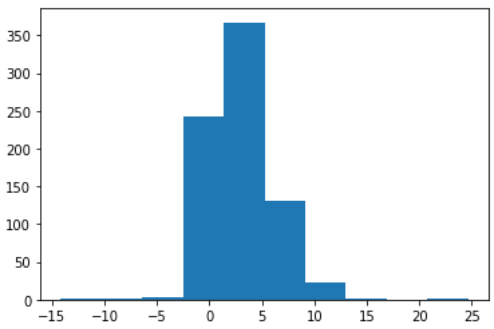

400
300
200
100
0
25
30
35
40
45
50
55
60


500
400
300
200
100
0
−120
−100
−80
−60
−40
−20
0
20


600
500
400
300
200
100
0
0.0
0.2
0.4
0.6
0.8
1.0


500
400
300
200
100
0
−10
−5
0
5
10
15
20


400
350
300
250
200
150
100
50
0
−10
−5
0
5
10
15
20


350
300
250
200
150
100
50
0
−15
−10
−5
0
5
10
15
20
25


300
250

7. Show all the categorical columns and save it to a new dataframe. (2pt)

```
#TODO
categorical_features = df.select_dtypes(include=[np.object])
categorical_features.columns
```

```
<ipython-input-9-e51306b70949>:2: DeprecationWarning: `np.object` is a deprecated alias for the builtin `object`. To silence this warning, use `
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  categorical_features = df.select_dtypes(include=[np.object])
Index(['state_code', 'zip_code', 'id', 'city', 'name', 'founded_at',
       'closed_at', 'first_funding_at', 'last_funding_at', 'category_code',
       'status'],
      dtype='object')
```

8. Examine missing values (2+2+3=7pt)

8.1 Show a list with column wise count of missing values and display the list in count wise descending order.

```
#TODO
nulls = df.isnull().sum().to_frame('nulls')
nulls.sort_values("nulls", inplace = True, ascending = False)
nulls
```

	nulls
closed_at	588
age_last_milestone_year	152
age_first_milestone_year	152
state_code	0
age_last_funding_year	0
is_top500	0
avg_participants	0
category_code	0
milestones	0
funding_total_usd	0
funding_rounds	0
relationships	0
age_first_funding_year	0
latitude	0
last_funding_at	0
first_funding_at	0
founded_at	0
labels	0
name	0
city	0
id	0
zip_code	0
longitude	0
status	0

8.2 Show columnwise percentage of missing values.

```
#TODO
percentage = df.isnull().mean()*100
percentage = percentage.to_frame("nulls")
percentage.sort_values("nulls", inplace = True)
```

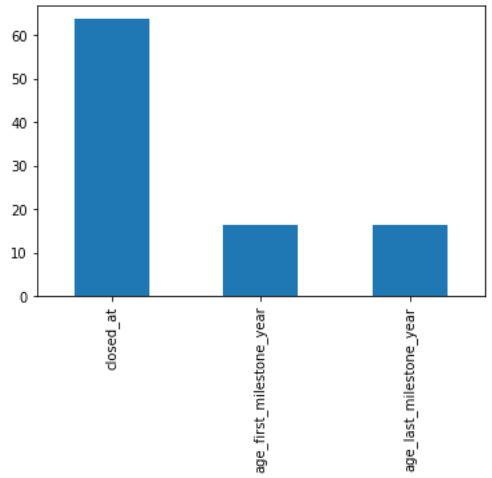
```
for index, row in percentage.iterrows():
    print(index, row[0])
```

```
state_code 0.0
avg_participants 0.0
category_code 0.0
milestones 0.0
funding_total_usd 0.0
funding_rounds 0.0
relationships 0.0
age_last_funding_year 0.0
age_first_funding_year 0.0
is_top500 0.0
last_funding_at 0.0
founded_at 0.0
labels 0.0
name 0.0
city 0.0
id 0.0
zip_code 0.0
longitude 0.0
latitude 0.0
first_funding_at 0.0
status 0.0
age_first_milestone_year 16.468039003250272
age_last_milestone_year 16.468039003250272
closed_at 63.705308775731304
```

8.3 Display a bar plot to visualize only the columns with missing values and their percentage count.

```
#TODO
miss = df.isnull().sum()
missing = 100 * miss / len(df)
missing = missing[missing>0]
missing.plot.bar()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7efd4701a2b0>
```



9. Label Encoding : Copy the dataframe to a new one. Then using scikitlearn's Label Encoder, transform the "status" column to 0-1. (5pt)

```
#TODO
newdf = df
le = LabelEncoder()
newdf.status = le.fit_transform(df.status)
newdf.head
```

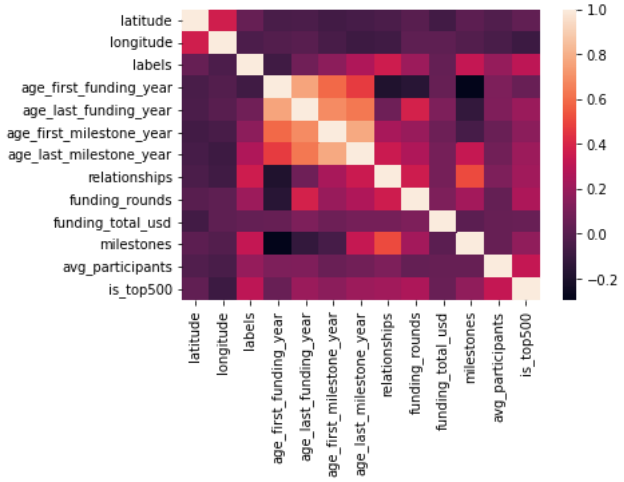
```
919         MA  42.504817  -71.195611     1803  c:41747     Burlington
920         CA  37.408261 -122.015920    94089  c:31549      Sunnyvale
921         CA  37.556732 -122.288378    94404  c:33198  San Francisco
922         CA  37.386778 -121.966277    95054  c:26702    Santa Clara

                    name  labels founded_at  closed_at  ...  \
```

```
920       Paracor Medical       0   1/1/1999  6/17/2012  ...
921              Causata       1   1/1/2009        NaN  ...
922  Asempra Technologies       1   1/1/2003        NaN  ...

    age_first_milestone_year  age_last_milestone_year  relationships  \
0                     4.6685                   6.7041              3
1                     7.0055                   7.0055              9
2                     1.4575                   2.2055              5
3                     6.0027                   6.0027              5
4                     0.0384                   0.0384              2
..                       ...                      ...            ...
918                   0.5808                   4.5260              9
919                   6.0027                   6.0027              1
920                   9.0055                   9.0055              5
921                   0.7589                   3.8356             12
922                   4.0027                   4.0027              4

     funding_rounds  funding_total_usd  milestones  category_code  \
0                 3             375000           3          music
1                 4           40100000           1     enterprise
2                 1            2600000           2            web
3                 3           40000000           1       software
4                 2            1300000           1    games_video
..              ...                ...         ...            ...
918               1            1100000           2    advertising
919               3           52000000           1       security
920               1           44000000           1        biotech
921               2           15500000           2       software
922               1           20000000           1       security

     avg_participants  is_top500  status
0              1.0000          0       0
1              4.7500          1       0
2              4.0000          1       0
3              3.3333          1       0
4              1.0000          1       1
..                ...        ...     ...
918            6.0000          1       0
919            2.6667          1       1
920            8.0000          1       1
921            1.0000          1       0
922            3.0000          1       0

[923 rows x 24 columns]>
```

10. Correlation: Use seaborn's heatmap to visualize the correlation between numeric features. (3pt)

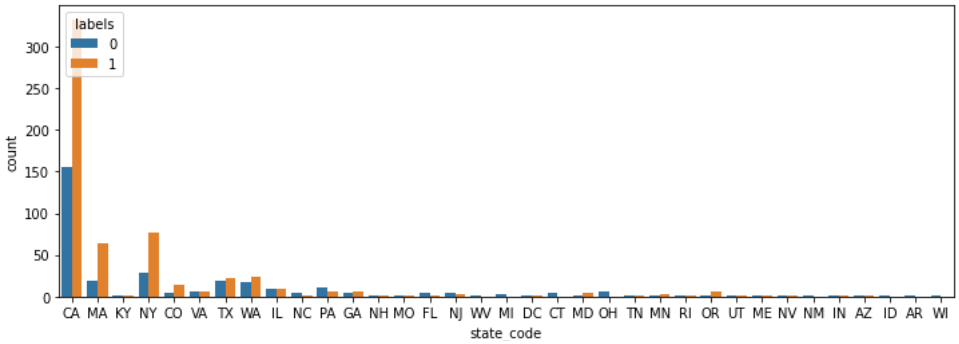
```
#TODO
sns.heatmap(numeric_features.corr())
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7efd471a2d00>
```



11.1 Use seaborn's countplot to visualize relationship between "*state_code*" and "*labels*". Comment on which state produced majority of successful startups (3pt)

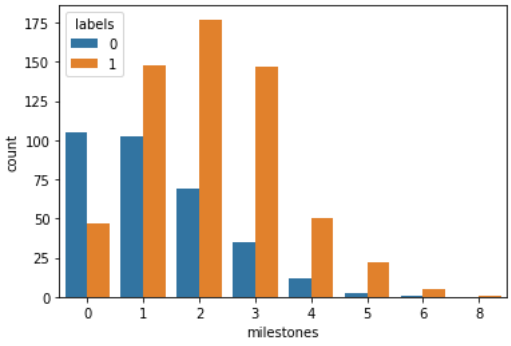
```
#TODO
fig = plt.figure(figsize=(12,4))
sns.countplot(x = df['state_code'], hue = df['labels'], data = df)
plt.show()
#CA produced majority of successful startups
```



11.2 Use seaborn's countplot to visualize relationship between "*milestones*" and "*labels*". Comment on which milestone made the statistically highest number of successful startups (3pt)

Double-click (or enter) to edit

```
#TODO
sns.countplot(x = df['milestones'], hue = df['labels'], data = df)
plt.show()
#2 milestones made the statistically highest number of successful startups
```



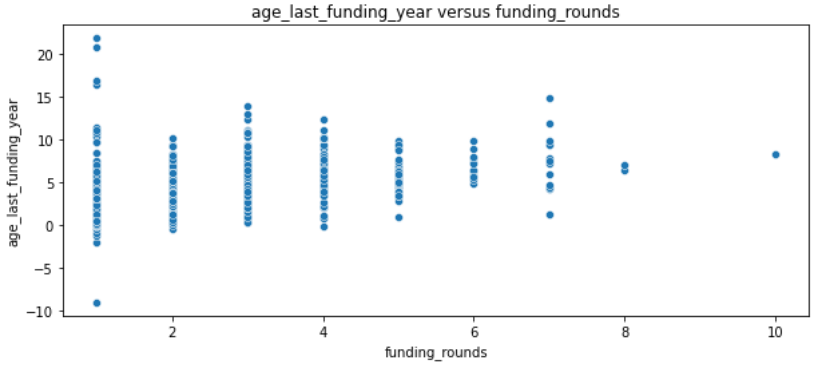
12. Drop features with duplicate values in-place, then show dataframe's new shape. (1pt)

```
#TODO
df.drop_duplicates(inplace=True)
df.shape
```

```
(923, 24)
```

13. From correlation heatmap above, comment on which feature has the highest correlation with "*funding_rounds*". Visualize a scatterplot with that and "*funding_rounds*". (3+3 = 6pt)

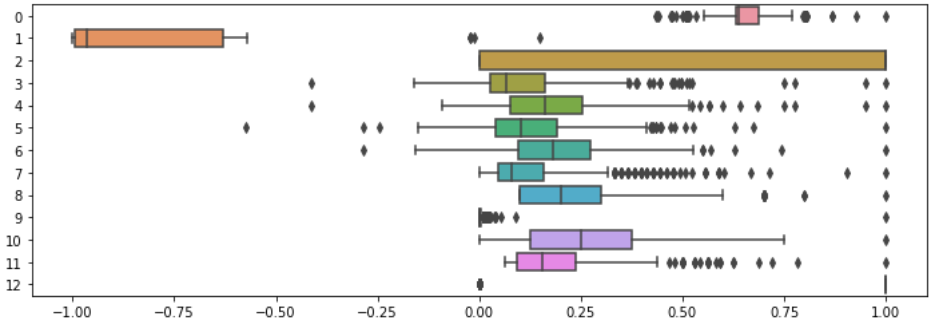
```
#TODO
#The 'age_last_funding_year' has the highest correlation with 'funding_rounds'
plt.figure(figsize = (10,4))
sns.scatterplot(data = df, x = 'funding_rounds', y = 'age_last_funding_year').set(title = 'age_last_funding_year versus funding_rounds')
plt.show()
```



14. Show boxplots for the numeric features to detect outliers. (4pt)

```
#TODO
fig = plt.figure(figsize=(12,4))
nf2 = numeric_features
for i in nf2:
    nf2[i] = nf2[i] / nf2[i].abs().max()
sns.boxplot(data = [nf2['latitude'],nf2['longitude'],nf2['labels'],
            nf2['age_first_funding_year'],nf2['age_last_funding_year'],
            nf2['age_first_milestone_year'],nf2['age_last_milestone_year'],
            nf2['relationships'],nf2['funding_rounds'],
            nf2['funding_total_usd'],nf2['milestones'],nf2['avg_participants'],
            nf2['is_top500']], orient = "h")
plt.show()



"""
sns.boxplot(numeric_features['latitude'])
plt.show()
sns.boxplot(numeric_features['longitude'])
plt.show()
sns.boxplot(numeric_features['labels'])
plt.show()
sns.boxplot(numeric_features['age_first_funding_year'])
plt.show()
sns.boxplot(numeric_features['age_last_funding_year'])
plt.show()
sns.boxplot(numeric_features['age_first_milestone_year'])
plt.show()
sns.boxplot(numeric_features['age_last_milestone_year'])
plt.show()
sns.boxplot(numeric_features['relationships'])
plt.show()
sns.boxplot(numeric_features['funding_rounds'])
plt.show()
sns.boxplot(numeric_features['funding_total_usd'])
plt.show()
sns.boxplot(numeric_features['milestones'])
plt.show()
sns.boxplot(numeric_features['avg_participants'])
plt.show()
sns.boxplot(numeric_features['is_top500'])
plt.show()
"""
```

```
<ipython-input-20-c2698eadfc1f>: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-c
  nf2[i] = nf2[i] / nf2[i].abs().max()
```




```
'\nsns.boxplot(numeric_features['latitude'])\nplt.show()\nsns.boxplot(numeric_features['longitude'])\nplt.show()\nsns.boxplot(numeric_features['labels'])\nplt.show()\nsns.boxplot(numeric_features['age_first_funding_year'])\nplt.show()\nsns.boxplot(numeric_features['age_last_funding_year'])\nplt.show()\nsns.boxplot(numeric_features['age_first_milestone_year'])\nplt.show()\nsns.boxplot(numeric_features['age_last_milestone_year'])\nplt.show()\nsns.boxplot(numeric_features['relationships'])\nplt.show()\nsns.boxplot(numeric_features['funding_rounds'])\nplt.show()\nsns.boxplot(numeric_features['funding_total_usd'])\nplt.show()\nsns.boxplot(numeric_features['milestones'])
```

15. Summary and Discussion: Mention what additional steps are required to use this dataset in a binary classifier. Eg: any column to remove, any record to remove, any distribution to rebalance, any features to be joined together to generate new feature etc. (2pt)

Several additional steps:

1. 'latitude' and 'longitude' can be combined as one feature
2. Use Simple Random Sampling to extract key data
3. Use k means to cluster dataset
4. Use normalization to compress values into a small range