```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
import warnings
warnings.filterwarnings("ignore")
```

```
startup_data = pd.read_csv("/content/startup_data.csv")
```

```
startup_data
```

	Unnamed: 0	state_code	latitude	longitude	zip_code	id	city	Unnamed
0	1005	CA	42.358880	-71.056820	92101	c:6669	San Diego	Na
1	204	CA	37.238916	-121.973718	95032	c:16283	Los Gatos	Na
2	1001	CA	32.901049	-117.192656	92121	c:65620	San Diego	Sa Diego C 921
3	738	CA	37.320309	-122.050040	95014	c:42668	Cupertino	Cupertin CA 950
4	1002	CA	37.779281	-122.419236	94105	c:65806	San Francisco	Sa Francis CA 941
...	...	...	...	...	...	...	...	
918	352	CA	37.740594	-122.376471	94107	c:21343	San Francisco	Na
919	721	MA	42.504817	-71.195611	1803	c:41747	Burlington	Burlingt MA 18
920	557	CA	37.408261	-122.015920	94089	c:31549	Sunnyvale	Na
921	589	CA	37.556732	-122.288378	94404	c:33198	San Francisco	Na
922	462	CA	37.386778	-121.966277	95054	c:26702	Santa Clara	San Clara C 950

923 rows × 49 columns

```
startup_data.shape
```

```
(923, 49)
```

```
startup_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 923 entries, 0 to 922
Data columns (total 49 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Unnamed: 0                923 non-null    int64
 1   state_code                923 non-null    object
 2   latitude                  923 non-null    float64
 3   longitude                 923 non-null    float64
 4   zip_code                  923 non-null    object
 5   id                        923 non-null    object
 6   city                      923 non-null    object
 7   Unnamed: 6                430 non-null    object
 8   name                      923 non-null    object
 9   labels                    923 non-null    int64
 10  founded_at                923 non-null    object
 11  closed_at                 335 non-null    object
 12  first_funding_at          923 non-null    object
 13  last_funding_at           923 non-null    object
 14  age_first_funding_year    923 non-null    float64
 15  age_last_funding_year     923 non-null    float64
 16  age_first_milestone_year  771 non-null    float64
 17  age_last_milestone_year   771 non-null    float64
 18  relationships             923 non-null    int64
```

```
 19  funding_rounds            923 non-null    int64
 20  funding_total_usd         923 non-null    int64
 21  milestones                923 non-null    int64
 22  state_code.1              922 non-null    object
 23  is_CA                     923 non-null    int64
 24  is_NY                     923 non-null    int64
 25  is_MA                     923 non-null    int64
 26  is_TX                     923 non-null    int64
 27  is_otherstate             923 non-null    int64
 28  category_code             923 non-null    object
 29  is_software               923 non-null    int64
 30  is_web                    923 non-null    int64
 31  is_mobile                 923 non-null    int64
 32  is_enterprise             923 non-null    int64
 33  is_advertising            923 non-null    int64
 34  is_gamesvideo             923 non-null    int64
 35  is_ecommerce              923 non-null    int64
 36  is_biotech                923 non-null    int64
 37  is_consulting             923 non-null    int64
 38  is_othercategory          923 non-null    int64
 39  object_id                 923 non-null    object
 40  has_VC                    923 non-null    int64
 41  has_angel                 923 non-null    int64
 42  has_roundA                923 non-null    int64
 43  has_roundB                923 non-null    int64
 44  has_roundC                923 non-null    int64
 45  has_roundD                923 non-null    int64
 46  avg_participants          923 non-null    float64
 47  is_top500                 923 non-null    int64
 48  status                    923 non-null    object
dtypes: float64(7), int64(28), object(14)
memory usage: 353.5+ KB
```

```
startup_data.describe()
```

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

8 rows × 35 columns

```
# Select only numeric columns
numeric_columns = startup_data.select_dtypes(include=['float64', 'int64'])

# Calculate correlation
correlation_matrix = numeric_columns.corr()
```

```
startup_data.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', 'is_CA', 'is_NY',
       'is_MA', 'is_TX', 'is_otherstate', 'category_code', 'is_software',
       'is_web', 'is_mobile', 'is_enterprise', 'is_advertising',
       'is_gamesvideo', 'is_ecommerce', 'is_biotech', 'is_consulting',
       'is_othercategory', 'object_id', 'has_VC', 'has_angel', 'has_roundA',
       'has_roundB', 'has_roundC', 'has_roundD', 'avg_participants',
       'is_top500', 'status'],
      dtype='object')
```

```
# Drop the irrelevant features
columns_to_drop = ['state_code', 'latitude', 'longitude', 'id', 'Unnamed: 6', 'state_code.1', 'zip_code', 'founded_at', 'closed_at', 'f

# Drop columns
startup_data.drop(columns_to_drop, axis=1, inplace=True)
```

```
# Checking if there are null values or not

startup_data.isnull().sum()
```

```
Unnamed: 0                   0
city                         0
labels                       0
age_first_funding_year       0
age_last_funding_year        0
age_first_milestone_year   152
age_last_milestone_year    152
relationships                0
funding_rounds               0
funding_total_usd            0
milestones                   0
is_CA                        0
is_NY                        0
is_MA                        0
is_TX                        0
is_otherstate                0
category_code                0
is_software                  0
is_web                       0
is_mobile                    0
is_enterprise                0
is_advertising               0
is_gamesvideo                0
is_ecommerce                 0
is_biotech                   0
is_consulting                0
is_othercategory             0
has_VC                       0
has_angel                    0
has_roundA                   0
has_roundB                   0
has_roundC                   0
has_roundD                   0
avg_participants             0
is_top500                    0
status                       0
dtype: int64
```

```
# Filling null values

startup_data['age_first_milestone_year'] = startup_data['age_first_milestone_year'].fillna(startup_data['age_first_milestone_year'].mear
startup_data['age_last_milestone_year'] = startup_data['age_last_milestone_year'].fillna(startup_data['age_last_milestone_year'].mean()
```

```
# Data after filling null values:

startup_data.isnull().sum()
```

```
Unnamed: 0                 0
city                       0
labels                     0
age_first_funding_year     0
age_last_funding_year      0
age_first_milestone_year   0
age_last_milestone_year    0
relationships              0
funding_rounds             0
funding_total_usd          0
milestones                 0
is_CA                      0
is_NY                      0
is_MA                      0
is_TX                      0
is_otherstate              0
category_code              0
is_software                0
is_web                     0
is_mobile                  0
is_enterprise              0
is_advertising             0
is_gamesvideo              0
is_ecommerce               0
is_biotech                 0
is_consulting              0
is_othercategory           0
has_VC                     0
has_angel                  0
has_roundA                 0
has_roundB                 0
has_roundC                 0
has_roundD                 0
avg_participants           0
is_top500                  0
```

```
status                    0
dtype: int64
```

```
# Taget variable is object so converting it into int:

encoder=LabelEncoder()
```

```
startup_data['status']= encoder.fit_transform(startup_data['status'])
startup_data
```

	Unnamed: 0	city	labels	age_first_funding_year	age_last_funding_year	age_f
0	1005	San Diego	1	2.2493	3.0027	
1	204	Los Gatos	1	5.1260	9.9973	
2	1001	San Diego	1	1.0329	1.0329	
3	738	Cupertino	1	3.1315	5.3151	
4	1002	San Francisco	0	0.0000	1.6685	
...	...	...	...	...	...	
918	352	San Francisco	1	0.5178	0.5178	
919	721	Burlington	0	7.2521	9.2274	
920	557	Sunnyvale	0	8.4959	8.4959	
921	589	San Francisco	1	0.7589	2.8329	
922	462	Santa Clara	1	3.1205	3.1205	

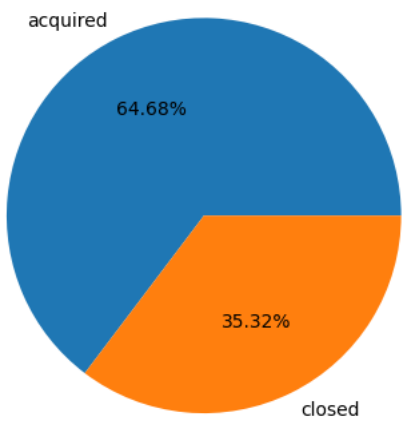
923 rows × 36 columns

```
# sample for acquired and closed

startup_data['status'].value_counts()
```

```
status
0    597
1    326
Name: count, dtype: int64
```

```
plt.pie(startup_data['status'].value_counts(), labels=['acquired','closed'], autopct='%2.2f%%')
plt.show()
```




```
# Top 10 cities for startup

startup_data['city'].value_counts().head(10)
```

```
city
San Francisco    128
New York          91
Mountain View     47
```

```
Palo Alto         35
Santa Clara       27
Austin            27
San Mateo         26
Seattle           26
Sunnyvale         22
San Jose          18
Name: count, dtype: int64
```

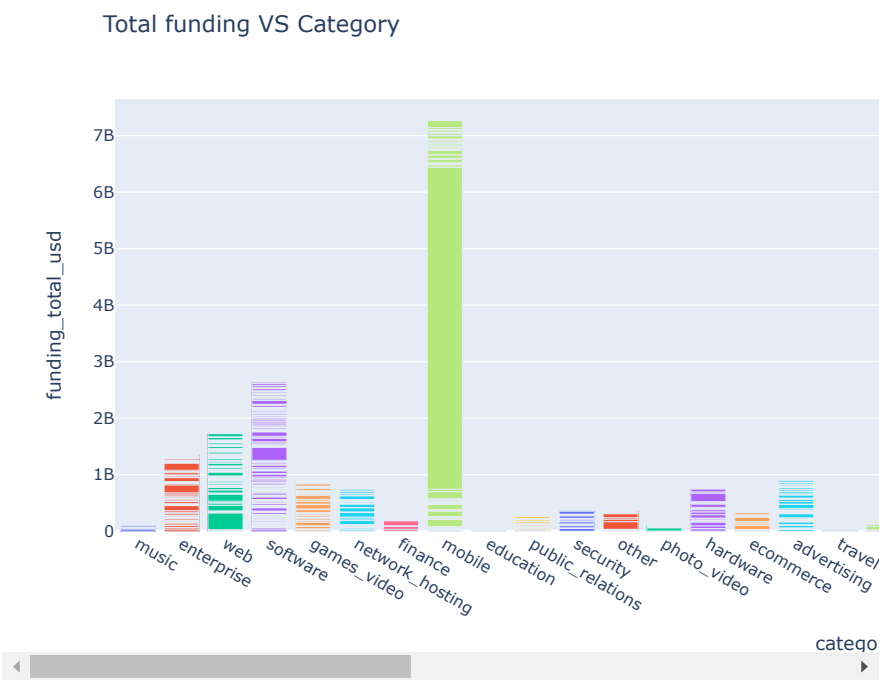
```
# Top 10 startup industries:

startup_data['category_code'].value_counts().head(10)
```

```
category_code
software           153
web                144
mobile              79
enterprise          73
advertising         62
games_video         52
semiconductor       35
network_hosting     34
biotech             34
hardware            27
Name: count, dtype: int64
```

```
# Graph of funding_total_usd Vs category_code

fig=px.bar(startup_data,x=startup_data['category_code'],y=startup_data['funding_total_usd'],title="Total funding VS Category",color='cat
fig.show()
```

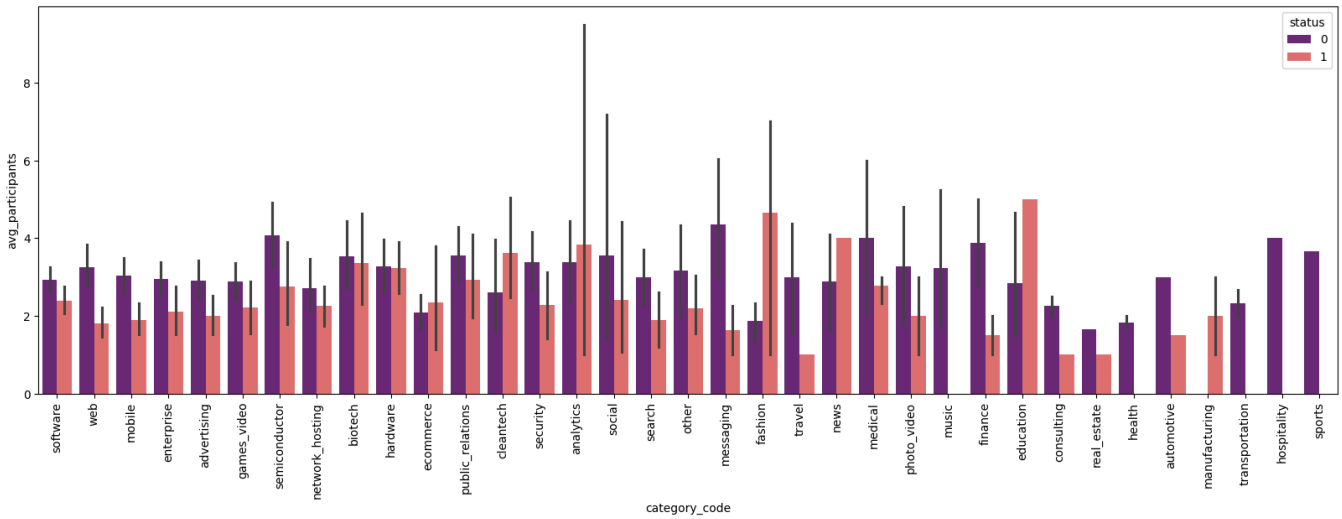



```
# Graph of avg_participants Vs category_code

fig, ax = plt.subplots(figsize=(20,6))

startup_category= sns.barplot(x="category_code", y='avg_participants', hue="status", data=startup_data, palette="magma",
               order=startup_data.category_code.value_counts().index)

startup_category = ax.set_xticklabels(ax.get_xticklabels(), rotation=90)
plt.show()
```




```
startup_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 923 entries, 0 to 922
Data columns (total 36 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Unnamed: 0                923 non-null    int64
 1   city                      923 non-null    object
 2   labels                    923 non-null    int64
 3   age_first_funding_year    923 non-null    float64
 4   age_last_funding_year     923 non-null    float64
 5   age_first_milestone_year  923 non-null    float64
 6   age_last_milestone_year   923 non-null    float64
 7   relationships             923 non-null    int64
 8   funding_rounds            923 non-null    int64
 9   funding_total_usd         923 non-null    int64
 10  milestones                923 non-null    int64
 11  is_CA                     923 non-null    int64
 12  is_NY                     923 non-null    int64
 13  is_MA                     923 non-null    int64
 14  is_TX                     923 non-null    int64
 15  is_otherstate             923 non-null    int64
 16  category_code             923 non-null    object
 17  is_software               923 non-null    int64
 18  is_web                    923 non-null    int64
 19  is_mobile                 923 non-null    int64
 20  is_enterprise             923 non-null    int64
 21  is_advertising            923 non-null    int64
 22  is_gamesvideo             923 non-null    int64
 23  is_ecommerce              923 non-null    int64
 24  is_biotech                923 non-null    int64
 25  is_consulting             923 non-null    int64
 26  is_othercategory          923 non-null    int64
 27  has_VC                    923 non-null    int64
 28  has_angel                 923 non-null    int64
 29  has_roundA                923 non-null    int64
 30  has_roundB                923 non-null    int64
 31  has_roundC                923 non-null    int64
 32  has_roundD                923 non-null    int64
 33  avg_participants          923 non-null    float64
 34  is_top500                 923 non-null    int64
 35  status                    923 non-null    int64
dtypes: float64(5), int64(29), object(2)
memory usage: 259.7+ KB
```

```
#creating dummy variables for the categorical features
startup_data['city']= encoder.fit_transform(startup_data['city'])
startup_data['category_code']= encoder.fit_transform(startup_data['category_code'])
```

separating the Dataset

```
x=startup_data.drop('status', axis=1)
y=startup_data['status']
```

Splitting the dataset into training and test data

```
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.2)
```

```
# Implementation of Logistic Regression

from sklearn.linear_model import LogisticRegression

lr = LogisticRegression()

lr.fit(x_train, y_train)

# prediction
y_pred_lr = lr.predict(x_test)

print("Accuracy of the test set: ", accuracy_score(y_test, y_pred_lr))
```

```
Accuracy of the test set:  0.6432432432432432
```

```
# implemenetation of KNN algorithm

from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n_neighbors=6)
knn.fit(x_train, y_train)

# prediction
y_pred_knn = knn.predict(x_test)

print("Accuracy of the test set: ", accuracy_score(y_test, y_pred_knn))
```

```
Accuracy of the test set:  0.6648648648648648
```

```
# Implementation of Decision Tree:
from sklearn.tree import DecisionTreeClassifier

tree= DecisionTreeClassifier(criterion='entropy', random_state=20)

#tree = DecisionTreeClassifier(random_state=23)
tree.fit(x_train, y_train)

# prediction
y_pred_tr = tree.predict(x_test)

print("Accuracy of the test set: ", accuracy_score(y_test, y_pred_tr))
```

```
Accuracy of the test set:  1.0
```

```
# Gathering accuracy score for each model

Accuracy_score = { 'Logistic Regression': { 'Accuracy_score':  accuracy_score(y_test, y_pred_lr)},
                   'K Nearest Neighbor': {'Accuracy_score':  accuracy_score(y_test, y_pred_knn)},
                   'Decision Tree': {'Accuracy_score': accuracy_score(y_test, y_pred_tr)}}

# Plotting comparsion of each model
Accuracy_score = pd.DataFrame(Accuracy_score)
Accuracy_score.plot(kind="barh",figsize=(10,4)).legend(loc='upper center', ncol=3, title="Machine Learning Model")
```

```
<matplotlib.legend.Legend at 0x7d1681f78fd0>
```