```
{
 "cells": [
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "d010a330",
   "metadata": {},
   "outputs": [],
   "source": [
    "import pandas as pd\n",
    "import numpy as np\n",
    "import seaborn as sns\n",
    "import matplotlib.pyplot as plt\n",
    "from scipy import stats"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "f2d8eae1",
   "metadata": {},
   "outputs": [],
   "source": [
    "df= pd.read_csv(\"Toughest Sport by Skill_2.csv\")\n",
    "DF = pd.DataFrame(df)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "08a6b1ea",
   "metadata": {},
   "outputs": [],
   "source": [
    "print(DF.sum())\n",
    "print(DF.mean())\n",
    "print(DF.std())\n",
    "print(DF.mode()[DF.columns[0:6]])\n",
    "print(DF.median())\n",
    "print(DF.describe())\n",
    "print(DF.describe(include='all'))\n",
    "print(DF.head()[DF.columns[0:6]])\n",
    "print(np.var(DF))\n",
    "print(DF.head()[DF.columns[0:8]])"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "2f707d23",
```

```
 "metadata": {},
 "outputs": [],
 "source": [
  "df.Strength.tail()\n",
  "df.Popularity_Total.tail()"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "id": "d1821bae",
 "metadata": {},
 "outputs": [],
 "source": [
  "variance = np.var(df)\n",
  "print(variance)"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "id": "dcaeafe0",
 "metadata": {},
 "outputs": [],
 "source": [
  "np.corrcoef(df['Strength'], df['Total'])[0, 1]\n",
  "np.cov(df['Strength'], df['Total'])[0, 1]\n",
  "\n",
  "np.corrcoef(df['Agility'], df['Popularity_Total'])[0, 1]\n",
  "np.cov(df['Agility'], df['Popularity_Total'])[0, 1]\n",
  "\n",
  "np.corrcoef(df['Durability'], df['Popularity_Total'])[0, 1]\n",
  "np.cov(df['Durability'], df['Popularity_Total'])[0, 1]\n",
  "\n",
  "np.corrcoef(df['Analytical Aptitude'], df['Popularity_Total'])[0, 1]\n",
  "np.cov(df['Analytical Aptitude'], df['Popularity_Total'])[0, 1]\n",
  "\n",
  "np.corrcoef(df['Flexibility'], df['Popularity_Total'])[0, 1]\n",
  "np.cov(df['Flexibility'], df['Popularity_Total'])[0, 1]\n",
  "\n",
  "np.corrcoef(df['Endurance'], df['Popularity_Total'])[0, 1]\n",
  "np.cov(df['Endurance'], df['Popularity_Total'])[0, 1]"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "id": "4dc550df",
 "metadata": {},
 "outputs": [],
```

```
   "source": [
    "df.Strength.tail()\n",
    "df.Endurance.tail()\n",
    "df.Agility.tail()\n",
    "df['Analytical Aptitude'].tail()\n",
    "df.Flexibility.tail()\n",
    "df.Durability.tail()\n",
    "df.Popularity_Total.tail()"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "d5833638",
   "metadata": {},
   "outputs": [],
   "source": [
    "df.plot(kind='scatter', x='Strength', y='Total', color='Blue')\n",
    "plt.show()\n",
    "df.plot(kind='scatter', x='Endurance', y='Popularity_Total', color='Red')\n",
    "plt.show()\n",
    "df.plot(kind='scatter', x='Agility', y='Popularity_Total', color='Green')\n",
    "plt.show()\n",
    "df.plot(kind='scatter', x='Analytical Aptitude', y='Popularity_Total',
color='Purple')\n",
    "plt.show()\n",
    "df.plot(kind='scatter', x='Flexibility', y='Total', color='Red')\n",
    "plt.show()\n",
    "df.plot(kind='scatter', x='Durability', y='Popularity_Total', color='Teal')\n",
    "plt.show()\n",
    "df.plot(kind='scatter', x='Rank', y='Popularity_Total', color='Red')\n",
    "plt.show()\n",
    "df.plot(kind='scatter', x='Sport', y='Popularity_Total', color='Red')\n",
    "plt.show()  # hard to read but provides outlies\n",
    "df.groupby(['Strength', 'Speed']).size().unstack().plot(kind='bar',
stacked=True)\n",
    "plt.show()  # hard to read"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "9c31b54f",
   "metadata": {},
   "outputs": [],
   "source": [
    "df[['Strength']].plot(kind='hist', bins=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
rwidth=.8)\n",
    "plt.show()\n",
    "df[['Endurance']].plot(kind='hist', bins=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
```

```
rwidth=.8)\n",
    "plt.show()\n",
    "df[['Agility']].plot(kind='hist', bins=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
rwidth=.8)\n",
    "plt.show()\n",
    "df[['Durability']].plot(kind='hist', bins=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
rwidth=.8)\n",
    "plt.show()\n",
    "df[['Analytical Aptitude']].plot(kind='hist', bins=[0, 1, 2, 3, 4, 5, 6, 7, 8,
9, 10], rwidth=.8)\n",
    "plt.show()\n",
    "df[['Flexibility']].plot(kind='hist', bins=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
rwidth=.8)\n",
    "plt.show()\n",
    "df[['Nerve']].plot(kind='hist', bins=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
rwidth=.8)\n",
    "plt.show()\n",
    "df[['Popularity_Total']].plot(kind='hist', bins=[5, 10, 15, 20, 25, 30, 40, 50,
55, 60], rwidth=.9)\n",
    "plt.show()\n",
    "df[['Popularity']].plot(kind='hist', bins=[5, 10, 15, 20, 25, 30, 35, 40, 45,
50, 55, 60], rwidth=.9)\n",
    "plt.show()"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "ed3ddc18",
   "metadata": {},
   "outputs": [],
   "source": [
    "df['Popularity_Total'].value_counts().sort_index().plot.barh()  # hard to
read\n",

"df.groupby(\"Sport\").Popularity.mean().sort_values(ascending=False)[:5].plot.bar()
\n",

"df.groupby(\"Sport\").Popularity_Total.mean().sort_values(ascending=False)[:6].plot
.bar()  # good visual"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "f7164e86",
   "metadata": {},
   "outputs": [],
   "source": [
    "## Computing p-value between two variables\n",
```

```
    "# H0 :- means difference is 0\n",
    "# H1:- mean difference is not 0\n",
    "df[['Strength', 'Total']].describe()\n",
    "ttest, pval = stats.ttest_rel(df['Strength'], df['Total'])\n",
    "print(pval)\n",
    "if pval <= 0.05:\n",
    "    print(\"reject null hypothesis\")\n",
    "    print(\"accept alternate hypothesis\")\n",
    "else:\n",
    "    print(\"accept null hypothesis\")"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "54b95ccf",
   "metadata": {},
   "outputs": [],
   "source": [
    "## z-test\n",
    "import pandas as pd\n",
    "from scipy import stats\n",
    "from statsmodels.stats import weightstats as stests\n",
    "\n",
    "ztest, pval = stests.ztest(df['Agility'], x2=None, value=60)\n",
    "print(float(pval))\n",
    "if pval <= 0.05:\n",
    "    print(\"reject null hypothesis\")\n",
    "    print(\"accept alternate hypothesis\")\n",
    "else:\n",
    "    print(\"accept null hypothesis\")"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "b8817b3e",
   "metadata": {},
   "outputs": [],
   "source": [
    "## Multiple Regression\n",
    "import statsmodels.formula.api as smf\n",
    "\n",
    "formula = 'Strength ~ Total + Power + Popularity_Total'  # Analytical Aptitude
as a\n",
    "model = smf.ols(formula, data=df)  # function of Popularity Total\n",
    "results = model.fit()\n",
    "results.summary()\n",
    "inter = results.params['Intercept']\n",
    "slope = results.params['Popularity_Total']\n",
```

```
    "inter, slope\n",
    "slope_pvalue = results.pvalues['Popularity_Total']  # p-value of the slope
estimate\n",
    "slope_pvalue\n",
    "results.rsquared  # coefficient of determination"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "01fef20e",
   "metadata": {},
   "outputs": [],
   "source": [
    "## Logistic Regression, can't run since no target column had values between 0 &
1.\n",
    "formula = 'Popularity ~ Strength + Endurance'\n",
    "model = smf.logit(formula, data=df)\n",
    "results = model.fit()\n",
    "endog = pd.DataFrame(model.endog, columns=[model.endog_names])\n",
    "exog = pd.DataFrame(model.exog, columns=model.exog_names)\n",
    "results.summary()\n",
    "actual = endog['Popularity']\n",
    "baseline = actual.mean()\n",
    "baseline"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "4c490f84",
   "metadata": {},
   "outputs": [],
   "source": [
    "## PMF's\n",
    "probabilities = df['Strength'].value_counts(normalize=True, bins=range(1,
8))\n",
    "sns.barplot(probabilities.index, probabilities.values)\n",
    "_ = plt.xlabel('Srength')\n",
    "_ = plt.ylabel('PMF')\n",
    "plt.show()\n",
    "\n",
    "probabilities = df['Analytical Aptitude'].value_counts(normalize=True,
bins=range(1, 8))\n",
    "sns.barplot(probabilities.index, probabilities.values)\n",
    "_ = plt.xlabel('Analytical Aptitude')\n",
    "_ = plt.ylabel('PMF')\n",
    "plt.show()"
   ]
  },
```

```
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "6b5e6f34",
   "metadata": {},
   "outputs": [],
   "source": [
    "# PMF's Two Variables\n",
    "bins = np.arange(0, max(df['Strength']) + 1.5) - 0.5\n",
    "_ = plt.hist(df['Strength'], normed=True, bins=bins)\n",
    "_ = plt.hist(df['Popularity_Total'], normed=True, bins=bins)\n",
    "_ = plt.xlabel('Attribute')\n",
    "_ = plt.ylabel('PMF')\n",
    "plt.show()"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "e8953245",
   "metadata": {},
   "outputs": [],
   "source": [
    "# PMF's single variable\n",
    "bins = np.arange(0, max(df['Strength']) + 1.5) - 0.5\n",
    "_ = plt.hist(df['Strength'], normed=True, bins=bins)\n",
    "_ = plt.xlabel('Strength')\n",
    "_ = plt.ylabel('PMF')\n",
    "plt.show()\n",
    "\n",
    "bins = np.arange(0, max(df['Endurance']) + 1.5) - 0.5\n",
    "_ = plt.hist(df['Endurance'], normed=True, bins=bins)\n",
    "_ = plt.xlabel('Endurance')\n",
    "_ = plt.ylabel('PMF')\n",
    "plt.show()"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "665cf1c3",
   "metadata": {},
   "outputs": [],
   "source": [
    "## CDF's\n",
    "num_bins = 25\n",
    "counts, bin_edges = np.histogram(df['Strength'], bins=num_bins,
normed=True)\n",
    "cdf = np.cumsum(counts)\n",
    "plt.plot(bin_edges[1:], cdf / cdf[-1])\n",
```

```
    "_ = plt.xlabel('Strength')\n",
    "_ = plt.ylabel('CDF')\n",
    "## Shows the median and tail statistic for the variable\n",
    "for q in [50, 90, 95, 100]:\n",
    "    print(\"{}%% percentile: {}\".format(q, np.percentile(df['Strength'],
q)))\n",
    "    \n",
    "num_bins = 25\n",
    "counts, bin_edges = np.histogram(df['Endurance'], bins=num_bins,
normed=True)\n",
    "cdf = np.cumsum(counts)\n",
    "plt.plot(bin_edges[1:], cdf / cdf[-1])\n",
    "_ = plt.xlabel('Endurance')\n",
    "_ = plt.ylabel('CDF')\n",
    "## Shows the median and tail statistic for the variable\n",
    "for q in [50, 90, 95, 100]:\n",
    "    print(\"{}%% percentile: {}\".format(q, np.percentile(df['Strength'], q)))"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "4142c5c9",
   "metadata": {},
   "outputs": [],
   "source": [
    "## Analytical Distribution  probabilty Plot's\n",
    "import scipy.stats\n",
    "import numpy as np\n",
    "import matplotlib.pyplot as plt"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "036276b4",
   "metadata": {},
   "outputs": [],
   "source": [
    "data = df['Strength'] * np.random.randn(60) + 0.5\n",
    "counts, start, dx, _ = scipy.stats.cumfreq(data, numbins=25)\n",
    "x = np.arange(counts.size) * dx + start\n",
    "plt.plot(x, counts, 'ro')\n",
    "plt.xlabel('Strength')\n",
    "plt.ylabel('Cumulative Frequency')\n",
    "plt.title('Probability Plot')\n",
    "plt.show()"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "4142c5c9",
   "metadata": {},
   "outputs": [],
   "source": [
    "## Analytical Distribution  probabilty Plot's\n",
    "import scipy.stats\n",
    "import numpy as np\n",
    "import matplotlib.pyplot as plt"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "036276b4",
   "metadata": {},
   "outputs": [],
   "source": [
    "data = df['Strength'] * np.random.randn(60) + 0.5\n",
    "counts, start, dx, _ = scipy.stats.cumfreq(data, numbins=25)\n",
    "x = np.arange(counts.size) * dx + start\n",
    "plt.plot(x, counts, 'ro')\n",
    "plt.xlabel('Strength')\n",
    "plt.ylabel('Cumulative Frequency')\n",
    "plt.title('Probability Plot')\n",
    "plt.show()"
   ]
  },
  {
```

```
   "cell_type": "code",
   "execution_count": null,
   "id": "edbdf382",
   "metadata": {},
   "outputs": [],
   "source": [
    "## Hypothesis test's\n",
    "from scipy.stats import chi2_contingency\n",
    "\n",
    "data = df['Endurance']\n",
    "stat, p, dof, expected = chi2_contingency(data)\n",
    "print('stat=%.3f, p=%.1f' % (stat, p))\n",
    "if p > 0.05:\n",
    "    print('Probably independent')\n",
    "else:\n",
    "    print('Probably dependent')"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "96b140b4",
   "metadata": {},
   "outputs": [],
   "source": [
    "# ANOVA Hypothesis test\n",
    "from scipy.stats import f_oneway  # For use in project\n",
    "\n",
    "data1 = df['Strength']\n",
    "data2 = df['Total']\n",
    "# data3 = df['Rank']\n",
    "stat, p = f_oneway(data1, data2)\n",
    "print('stat=%.3f, p=%.3f' % (stat, p))\n",
    "if p <= 0.05:\n",
    "    print('Accept null hypothesis they are Probably the same distribution')\n",
    "else:\n",
    "    print('Reject the null hypothesis, they are Probably different
distributions')"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "82ab7949",
   "metadata": {},
   "outputs": [],
   "source": []
  },
  {
   "cell_type": "code",
```

```
   "execution_count": null,
   "id": "2e30ada2",
   "metadata": {},
   "outputs": [],
   "source": []
  }
 ],
 "metadata": {
  "kernelspec": {
   "display_name": "Python 3 (ipykernel)",
   "language": "python",
   "name": "python3"
  },
  "language_info": {
   "codemirror_mode": {
    "name": "ipython",
    "version": 3
   },
   "file_extension": ".py",
   "mimetype": "text/x-python",
   "name": "python",
   "nbconvert_exporter": "python",
   "pygments_lexer": "ipython3",
   "version": "3.9.7"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 5
}
```