```
{
 "cells": [
  {
   "cell_type": "code",
   "execution_count": 1,
   "metadata": {},
   "outputs": [
    {
     "ename": "SyntaxError",
     "evalue": "invalid syntax (<ipython-input-1-f67151940a27>, line 11)",
     "output_type": "error",
     "traceback": [
      "\u001b[1;36m  File \u001b[1;32m\"<ipython-input-1-f67151940a27>\"\u001b[1;36m, line \u001b[1;32m11\u001b[0m\n\u001b[1;33m    cd C:\\Users\\Dev\\Desktop\\Kaggle\\Salinity\u001b[0m\n\u001b[1;37m       ^\u001b[0m\n\u001b[1;31mSyntaxError\u001b[0m\u001b[1;31m:\u001b[0m invalid syntax\n"
     ]
    }
   ],
   "source": [
    "import numpy as np\n",
    "import pandas as pd\n",
    "import seaborn as sns\n",
    "import matplotlib.pyplot as plt\n",
    "from sklearn import preprocessing, svm\n",
    "from sklearn.model_selection import train_test_split\n",
    "from sklearn.linear_model import LinearRegression\n",
    "\n",
    "\n",
    "\n",
    "cd C:\\Users\\Dev\\Desktop\\Kaggle\\Salinity\n",
    "\n",
    "# Changing the file read location to the location of the dataset\n",
    "df = pd.read_csv('bottle.csv')\n",
    "df_binary = df[['Salnty', 'T_degC']]\n",
    "\n",
    "# Taking only the selected two attributes from the dataset\n",
    "df_binary.columns = ['Sal', 'Temp']\n",
    "\n",
    "# Renaming the columns for easier writing of the code\n",
    "df_binary.head()\n",
    "\n",
    "# Displaying only the 1st rows along with the column names\n",
    "\n",
    "\n",
    "sns.lmplot(x =\"Sal\", y =\"Temp\", data = df_binary, order = 2, ci = None)\n",
    "\n",
    "# Plotting the data scatter\n",
    "# Eliminating NaN or missing input numbers\n",
    "df_binary.fillna(method ='ffill', inplace = True)\n",
    "X = np.array(df_binary['Sal']).reshape(-1, 1)\n",
    "y = np.array(df_binary['Temp']).reshape(-1, 1)\n",
    "\n",
    "# Separating the data into independent and dependent variables\n",
    "# Converting each dataframe into a numpy array\n",
    "# since each dataframe contains only one column\n",
    "df_binary.dropna(inplace = True)\n",
    "\n",
    "# Dropping any rows with Nan values\n",
    "X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25)\n",
    "\n",
    "# Splitting the data into training and testing data\n",
    "regr = LinearRegression()\n",
    "\n",
    "regr.fit(X_train, y_train)\n",
    "print(regr.score(X_test, y_test))\n",
    "y_pred = regr.predict(X_test)\n",
    "plt.scatter(X_test, y_test, color ='b')\n",
    "plt.plot(X_test, y_pred, color ='k')\n",
    "\n",
    "plt.show()\n",
    "# Data scatter of predicted values\n",
    "df_binary500 = df_binary[:][:500]\n",
    "\n",
    "# Selecting the 1st 500 rows of the data\n",
    "sns.lmplot(x =\"Sal\", y =\"Temp\", data = df_binary500,\n",
    "\t\t\t\t\t\t\torder = 2, ci = None)\n",
    "df_binary500.fillna(method ='ffill', inplace = True)\n",
    "\n",
    "X = np.array(df_binary500['Sal']).reshape(-1, 1)\n",
    "y = np.array(df_binary500['Temp']).reshape(-1, 1)\n",
    "\n",
    "df_binary500.dropna(inplace = True)\n",
    "X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25)\n",
    "\n",
    "regr = LinearRegression()\n",
    "regr.fit(X_train, y_train)\n",
    "print(regr.score(X_test, y_test))\n",
    "y_pred = regr.predict(X_test)\n",
    "plt.scatter(X_test, y_test, color ='b')\n",
    "plt.plot(X_test, y_pred, color ='k')\n",
    "\n",
    "plt.show()\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {},
   "outputs": [],
   "source": []
  }
 ],
 "metadata": {
  "kernelspec": {
   "display_name": "Python 3",
   "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": "3.8.5"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 4
}
```