IBM SUPERVISED ML FINAL ASSIGNMENT

- Main objective of the analysis that specifies whether your model will be focused on prediction or interpretation.

  To predict the prices of medical insurance charges based on the given features with a high degree of accuracy using different types of regression models.

- Brief description of the data set you chose and a summary of its attributes.

  The dataset concerns the various attributes of patients across a region with their medical charges. The attributes include age, sex, BMI, no of children, smoking behaviour, region and the target label – charges.

```
In [2]: data.head()
```

Out[2]:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520

- Brief summary of data exploration and actions taken for data cleaning and feature engineering.

  Data exploration includes –

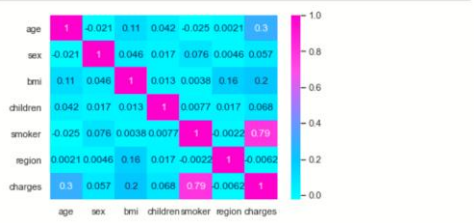
  - ➔ Checking null values – 0
  - ➔ Describing data
  - ➔ Label encoding categorial values – sex, smoker and region to analyse correlation and run the models
  - ➔ Correlation of different values – strong correlation seen between smoker and charges
  - ➔ Distribution plot of charges
  - ➔ Violin plots of charges vs sex and charges vs smoker
  - ➔ Regression plots of charges, age, bmi children – correlation also seen between charges with bmi and age

```
In [29]: data.head()
```

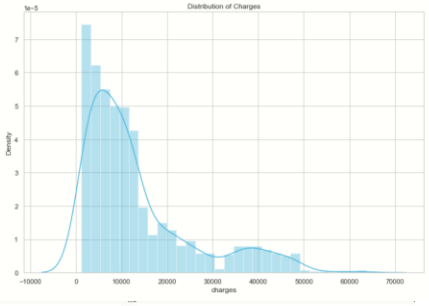
Out[29]:

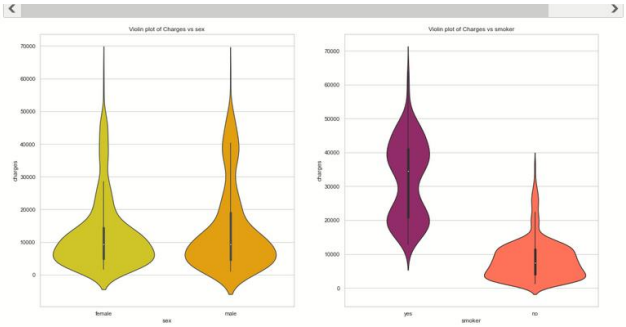
	age	sex	bmi	children	smoker	region	charges
0	19	0	27.900	0	1	3	16884.92400
1	18	1	33.770	1	0	2	1725.55230
2	28	1	33.000	3	0	2	4449.46200
3	33	1	22.705	0	0	1	21984.47061
4	32	1	28.880	0	0	1	3866.85520

```
In [30]: corr = data.corr()
         sns.heatmap(corr, cmap = 'cool', annot= True);
```

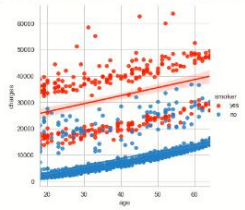


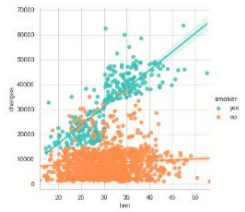
```
Out[16]: Text(0.5, 1.0, 'Distribution of Charges')
```

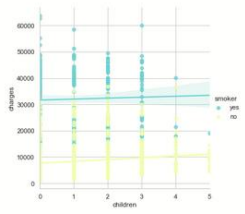





```
In [26]: ax = sns.lmplot(x = 'age', y = 'charges', data=data, hue='smoker', palette='Set1')
         ax = sns.lmplot(x = 'bmi', y = 'charges', data=data, hue='smoker', palette='Set2')
         ax = sns.lmplot(x = 'children', y = 'charges', data=data, hue='smoker', palette='Set3')
```







- Summary of training at least three linear regression models which should be variations that cover using a simple linear regression as a baseline, adding polynomial effects, and using a regularization regression. Preferably, all use the same training and test splits, or the same cross-validation method.
  - ➔ Among employed models – linear, lasso, ridge then adding polynomial features, polynomial features with linear regression give us the best result for this data.
  - ➔ All models using train test split

```
In [32]: from sklearn.model_selection import train_test_split as holdout
         from sklearn.linear_model import LinearRegression
         from sklearn import metrics
         x = data.drop(['charges'], axis = 1)
         y = data['charges']
         x_train, x_test, y_train, y_test = holdout(x, y, test_size=0.2, random_st
         Lin_reg = LinearRegression()
         Lin_reg.fit(x_train, y_train)
         print(Lin_reg.intercept_)
         print(Lin_reg.coef_)
         print(Lin_reg.score(x_test, y_test))
```

```
-11661.98390882441
[  253.99185244   -24.32455098   328.40261701   443.72929547
 23568.87948381  -288.50857254]
0.7998747145449959
```

```
In [33]: from sklearn.linear_model import Lasso
         Lasso = Lasso(alpha=0.2, fit_intercept=True, normalize=False, precompute=
                       tol=0.0001, warm_start=False, positive=False, random_state=
         Lasso.fit(x_train, y_train)
         print(Lasso.intercept_)
         print(Lasso.coef_)
         print(Lasso.score(x_test, y_test))
```

```
-11661.838929039537
[ 2.53991436e+02 -2.34569821e+01  3.28389438e+02  4.43587436e+02
  2.35676136e+04 -2.88340296e+02]
0.7998690236224705
```

```
In [34]: from sklearn.linear_model import Ridge
         Ridge = Ridge(alpha=0.5)
         Ridge.fit(x_train, y_train)
         print(Ridge.intercept_)
         print(Ridge.coef_)
         print(Ridge.score(x_test, y_test))
```

```
-11643.440927495807
[ 2.53893751e+02 -2.15112284e+01  3.28339566e+02  4.44238477e+02
  2.35009674e+04 -2.89027871e+02]
0.7996989632063138
```

```
In [41]: from sklearn.preprocessing import PolynomialFeatures
         x = data.drop(['charges', 'sex', 'region'], axis = 1)
         y = data.charges
         pol = PolynomialFeatures (degree = 2)
         x_pol = pol.fit_transform(x)
         x_train, x_test, y_train, y_test = holdout(x_pol, y, test_size=0.2, rando
         Pol_reg = LinearRegression()
         Pol_reg.fit(x_train, y_train)
         y_train_pred = Pol_reg.predict(x_train)
         y_test_pred = Pol_reg.predict(x_test)
         print(Pol_reg.intercept_)
         print(Pol_reg.coef_)
         print(Pol_reg.score(x_test, y_test))
```

```
-5325.881705252248
[ 0.00000000e+00 -4.01606591e+01  5.23702019e+02  8.52025026e+02
 -9.52698471e+03  3.04430186e+00  1.84508369e+00  6.01720286e+00
  4.20849790e+00 -9.38983382e+00  3.81612289e+00  1.40840670e+03
 -1.45982790e+02 -4.46151855e+02 -9.52698471e+03]
0.8812595703345225
```

- A paragraph explaining which of your regressions you recommend as a final model that best fits your needs in terms of accuracy and explain ability.
    - ➔ Using linear regression with polynomial features gives us the best results and predicts the values with high accuracy. Thus this is the best model of the purpose of predicting the target variable in this dataset.
    - ➔ The performance metrics are given below

```
In [9]: print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_
        print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_te
        print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error
```

```
Mean Absolute Error: 2824.4950454776617
Mean Squared Error: 18895160.09878044
Root Mean Squared Error: 4346.856346692451
```

```
In [10]: y_test_pred = Pol_reg.predict(x_test)

         df = pd.DataFrame({'Actual': y_test, 'Predicted': y_test_pred})
         df
```

Out[10]:

	Actual	Predicted
578	9724.53000	12101.156323
610	8547.69130	10440.782266
569	45702.02235	48541.022951
1034	12950.07120	14140.067522
198	9644.25250	8636.235727
...	...	...
1084	15019.76005	16712.196281
726	6664.68595	8654.565461
1132	20709.02034	12372.050609
725	40932.42950	41465.617268
963	9500.57305	10941.780705

268 rows × 2 columns

- Summary Key Findings and Insights, which walks your reader through the main drivers of your model and insights from your data derived from your linear regression model.
    - ➔ The smoker attribute has a high weightage in the model followed by BMI and age according to our correlation results
    - ➔ The final model is overall able to generalise and predict the values thus avoiding overfitting and underfitting

- Suggestions for next steps in analyzing this data, which may include suggesting revisiting this model adding specific data features to achieve a better explanation or a better prediction.

    - ➔ The model is highly simplistic thus, implementing random forest regression, KNN clustering, elastic net might improve results.

- ➔ The current dataset has only a handful of attributes thus only give a limited scope for interpretation
- ➔ Using a more updated dataset with more numerous and recent values will yield better interpretation of how factors like smoking, bmi, age affects health and health costs.