Predictions of Medical Insurance Charges Using `sklearn` and `keras`

Steven Hans

March 6, 2021

Use case

- Healthcare costs are high, especially in US.
- Make sure someone is fairly charged. Or at least within reasonable range.

Data set

Dataset obtained from
`https://www.kaggle.com/mirichoi0218/insurance` (Medical Cost Personal Datasets).

- A .csv file (1338 rows, 7 columns)
- Columns: age, sex, bmi, children, smoker, region, charges (more on these later)

Data quality assessment

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

Data exploration (numerical correlations)

Correlation for the numerical columns (rounded to two decimal places):

	age	bmi	children	charges
age	1.00	0.11	0.04	0.30
bmi	0.11	1.00	0.01	0.20
children	0.04	0.01	1.00	0.07
charges	0.30	0.20	0.07	1.00

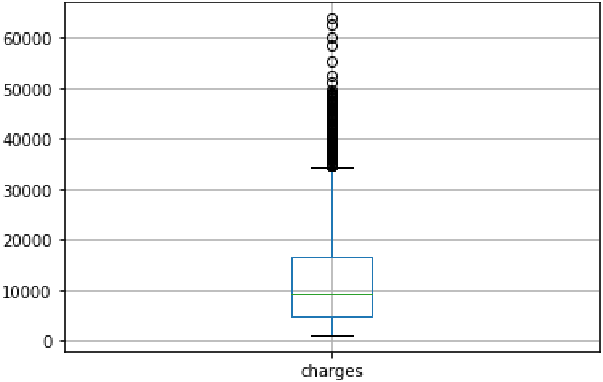
Observation: Number of children doesn't correlate much with charges.

Data exploration (statistics)

	age	bmi	children	charges
count	1338.000000	1338.000000	1338.000000	1338.000000
mean	39.207025	30.663397	1.094918	13270.422265
std	14.049960	6.098187	1.205493	12110.011237
min	18.000000	15.960000	0.000000	1121.873900
25%	27.000000	26.296250	0.000000	4740.287150
50%	39.000000	30.400000	1.000000	9382.033000
75%	51.000000	34.693750	2.000000	16639.912515
max	64.000000	53.130000	5.000000	63770.428010

- Maximum charge is really high. Probable sign of outliers.
- Standard deviation is pretty high. Could impact model performance.

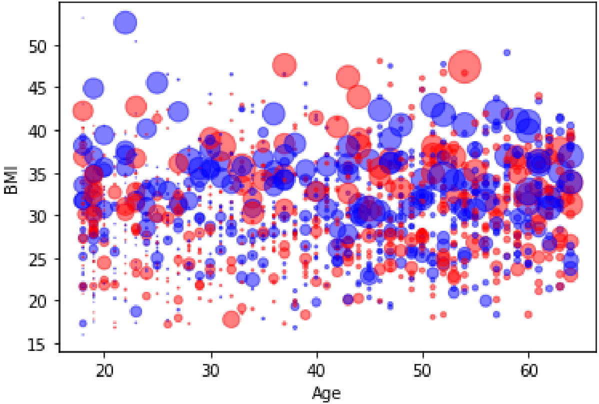
Data visualization (boxplot)



- Heavy tailed distributions.
- Outliers are part of the data & tells something about the distribution.

Data visualization (scatterplot 1)

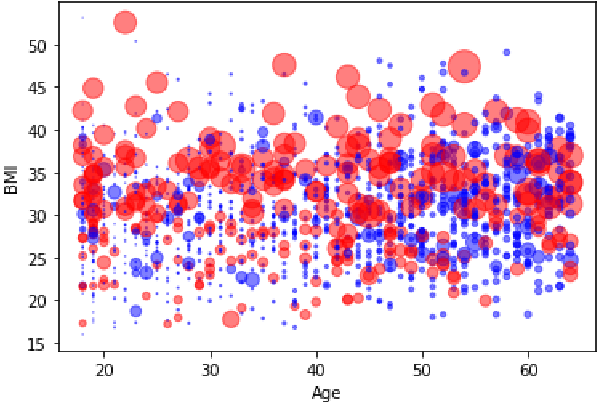
Assumption 1: There is a correlation between sex and the amount of medical charges.



Conclusion: No significant correlation.

Data visualization (scatterplot 2)

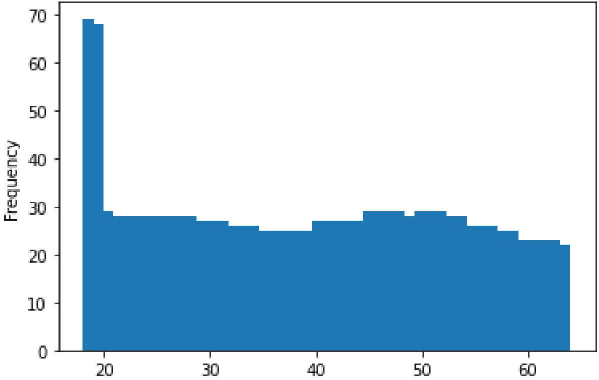
Assumption 2: There is a correlation between smoker and the amount of medical charges.



Conclusion: Strong correlation (smokers are charged more, indicated by larger circle size).

Data visualization (age histogram)

Observation: There seems to be a bunch of medical charge entries for age lower than 20.



- People buy insurance early for cheaper costs.
- Maybe people between 18-19 have more medical charges? Why the sudden drop?

Data preprocessing (cleaning the data)

Check for empty entries:

`insurance_dataset.isnull().sum()`

	null count
age	0
sex	0
bmi	0
children	0
smoker	0
region	0
charges	0

No empty entries. So far so good.

Data preprocessing (categorical columns)

Make sure there are no duplicate categories with different capitalization ('yes', 'yEs', 'YeS', 'YES')

```
insurance_dataset['sex'].unique()
insurance_dataset['smoker'].unique()
insurance_dataset['region'].unique()
```

Results:

```
array(['female', 'male'], dtype=object)
array(['yes', 'no'], dtype=object)
array(['southwest', 'southeast', 'northwest', 'northeast'],
dtype=object)
```

No duplicate categories.

Actual preprocessing step (StandardScaler, OneHotEncoder) using Pipeline

Drop the sex column, as it shows no significant correlation. Then, apply pipeline:

- StandardScaler: For numerical columns. Removing mean and scaling to unit variance helps model converge to optimal solution faster.
- OneHotEncoder: For categorical columns. Model can't handle texts as the input, so we one hot encode the columns.

We then split the data 75% train, 15% validation, 15% test.

MAE as the performance indicator

We want to penalize all errors equally, so we use mean absolute error.

$$MAE : \frac{1}{N} \sum_{i=1}^{N} |y_i - \hat{y}_i|$$

where N is the number of data, y_i is the actual value, and $\hat{y}_i$ is the predicted value.

Model 1: Naive model

This model always "predicts" the mean. This model serves as the baseline error.

```
train_mean = train['charges'].mean()
mae = (abs(train['charges']-train_mean)).sum()/len(train)

print('naive mae:', mae)
```

Result:

```
naive mae: 8861.963885950448
```

If our model can't even beat this, then we know something is wrong.

Model 2: `sklearn` model

- RandomForestRegressor is choosen to minimize overfitting & improve overall model accuracy.
- Use GridSearchCV to automate the best possible parameters available for the model.

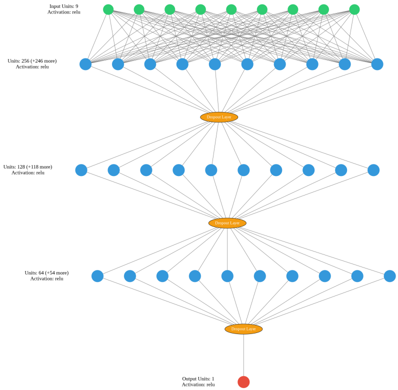
```
grid = [{'n_estimators': [2, 16, 64, 128],
       'max_features': [2, 8, 9]}]

model = RandomForestRegressor(random_state=1)
scoring='neg_mean_absolute_error',
return_train_score=True)

grid_search.fit(X_train, y_train)
```

Obtained 2764 MAE for 9 `max_features` and 64 `n_estimators`.

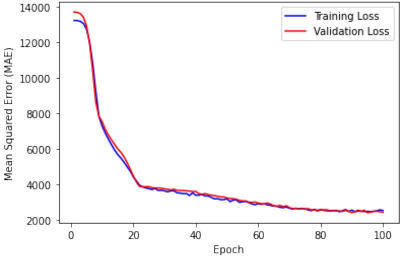
Model 3: `keras` model



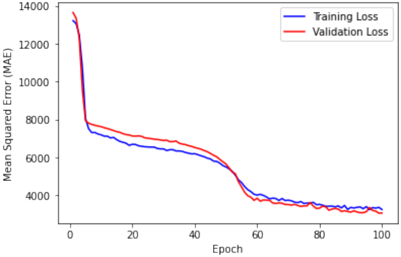
- Input layer with 9 units
- Dense layer with 256 units
- Dropout layer with 0.3 chance of randomly dropping out certain neurons
- Dense layer with 128 units
- Dropout layer with 0.2 chance
- Dense layer with 64 units
- Dropout layer with 0.1 chance
- Final dense layer predicting the charges

Model evaluations

Keras evaluation graph



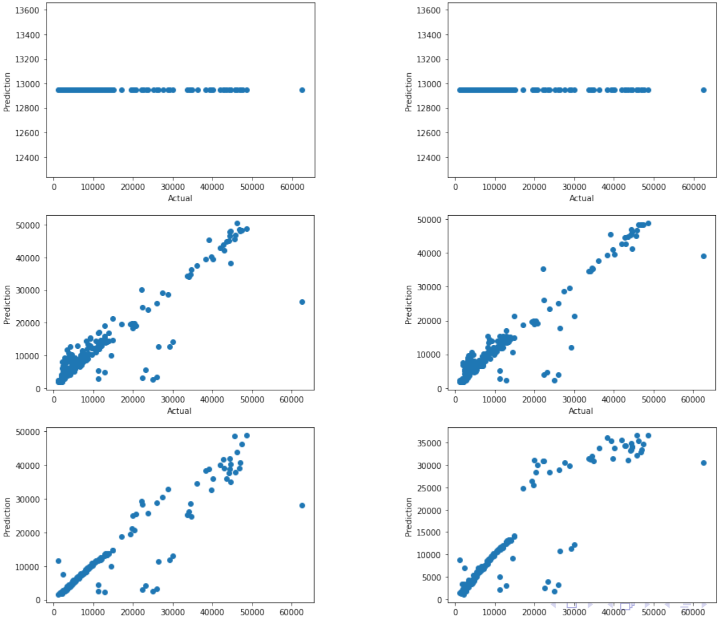
With feature engineering



Without feature engineering

Model evaluations

Scatterplot predictions for all three models:



Conclusion

- MAE for each final model on test set (lower is better):

Model	Mean Absolute Error (MAE)
Naive	9872
sklearn	2875
Keras	2064

  I'm choosing the Keras model because it generalises much better and therefore performs much better on unseen data.
- Building baseline model is important
- More data will improve model prediction. Additional column may help prediction (like hobbies)

Thank you!