IronKaggle

Sales prediction

Natanael Santiago Morales

Introduction

Sales prediction

Data set description:

- 640,840 rows
- 10 columns
    - Index, store_ID
    - Date, day of week
    - # of customers on the day
    - Open, promotion, state_holiday, school_holiday
    - Sales
- No empty rows or NaNs

Data Cleaning

Dropped

- Index (unnamed)
- Date

One-hot encoded

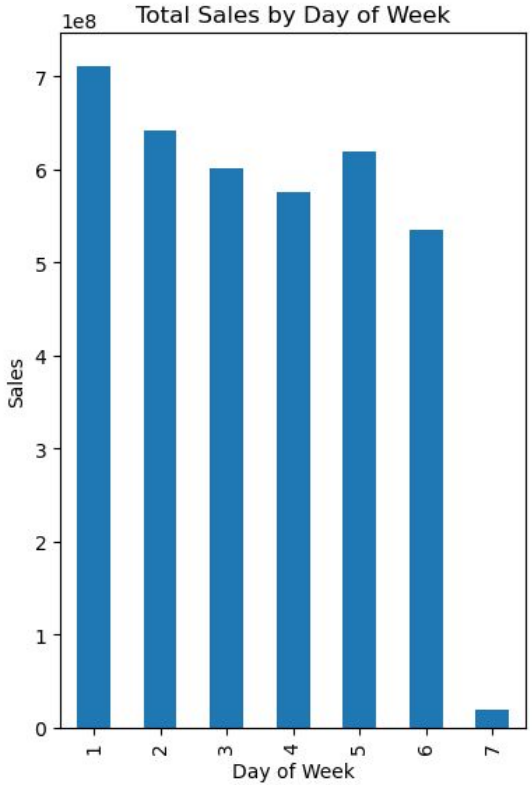
- State holiday
- Day of week

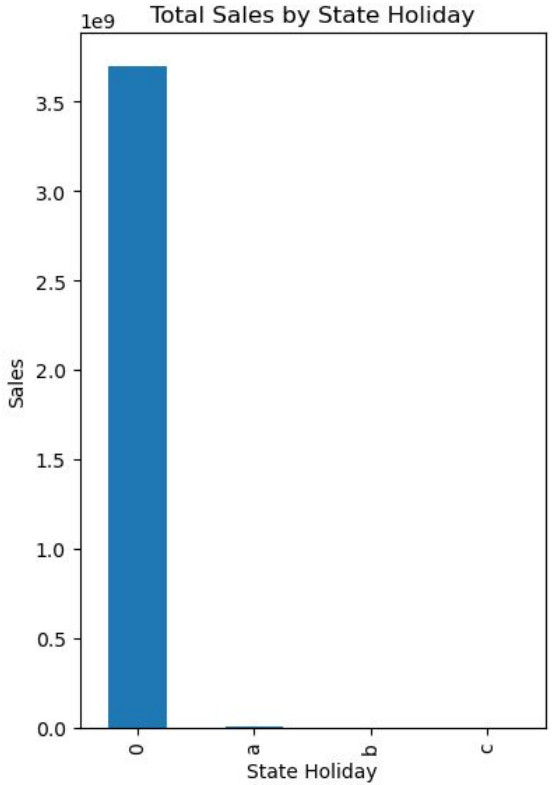
Engineered

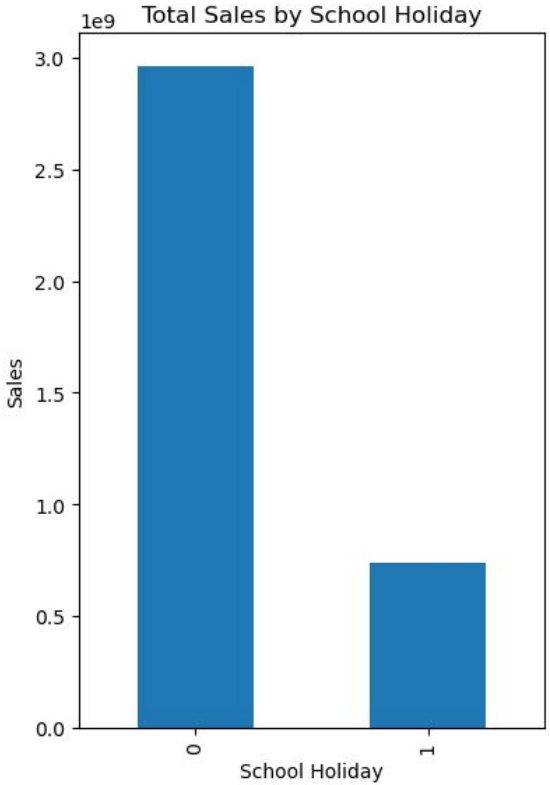
- weighted_factor

```
store_customers =
(X_train.groupby('store_ID')['nb_customers_on_day'].sum()/(X_train['nb_customers_on_day'].sum()))
store_occurrence = X_train['store_ID'].value_counts().sort_index()
weighted_factor = store_customers * store_occurrence
X_train['weighted_factor'] = X_train['store_ID'].map(weighted_factor)
```

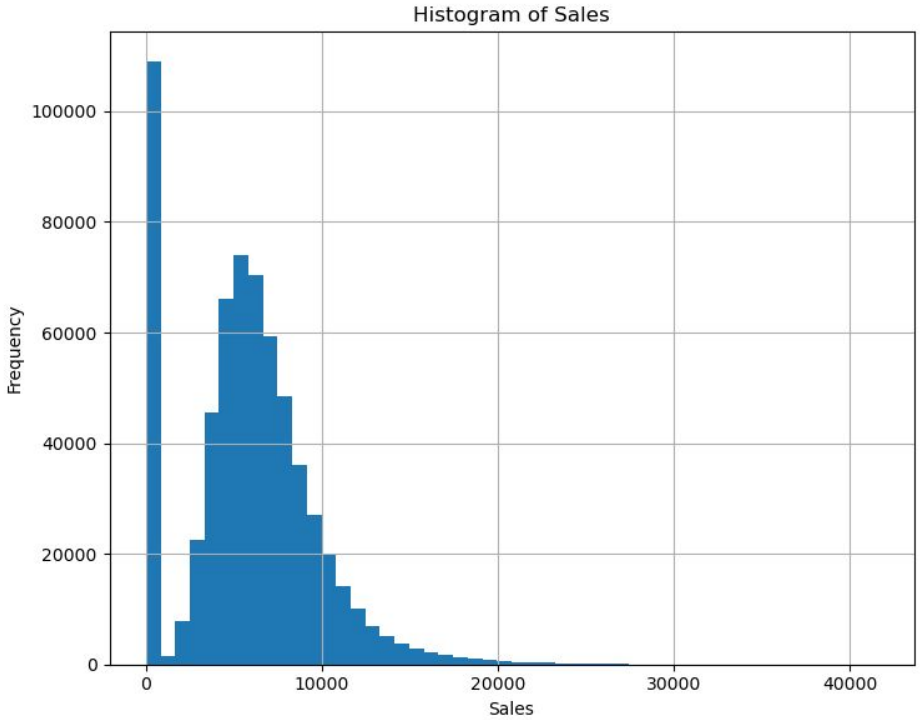
Data exploration

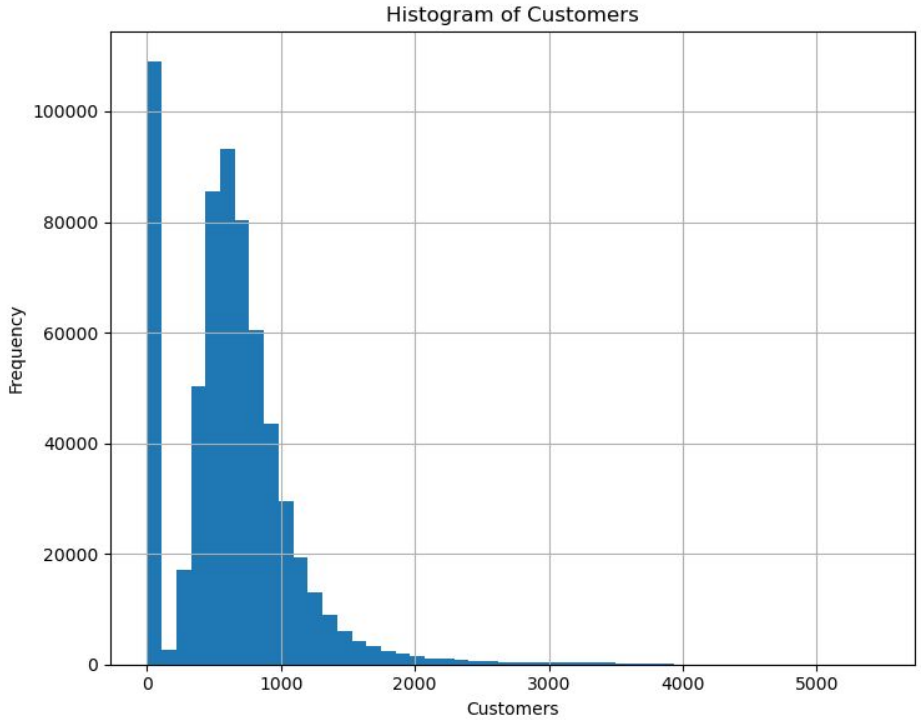




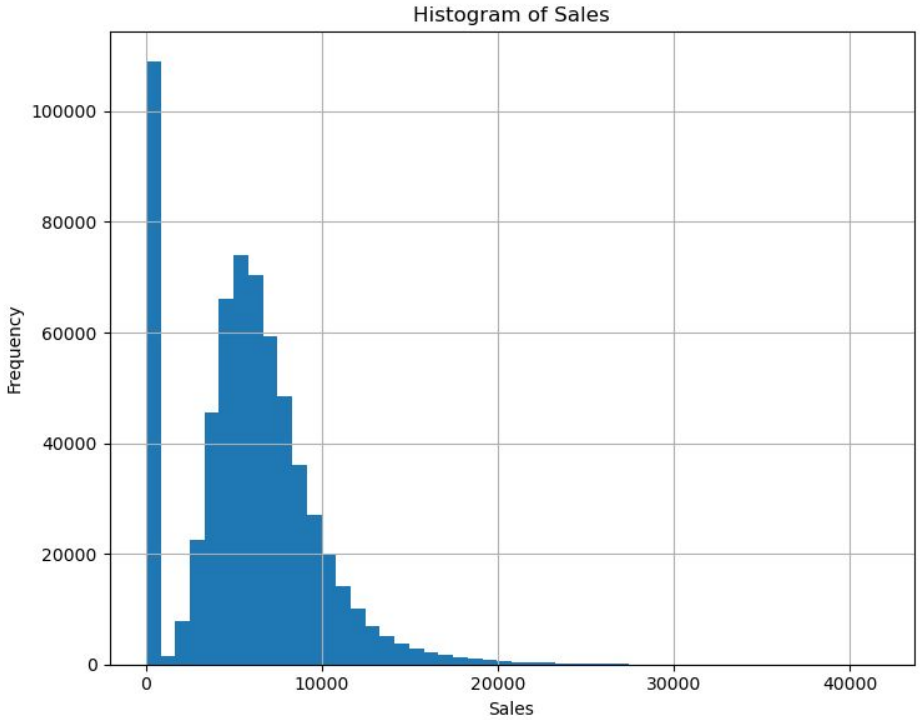


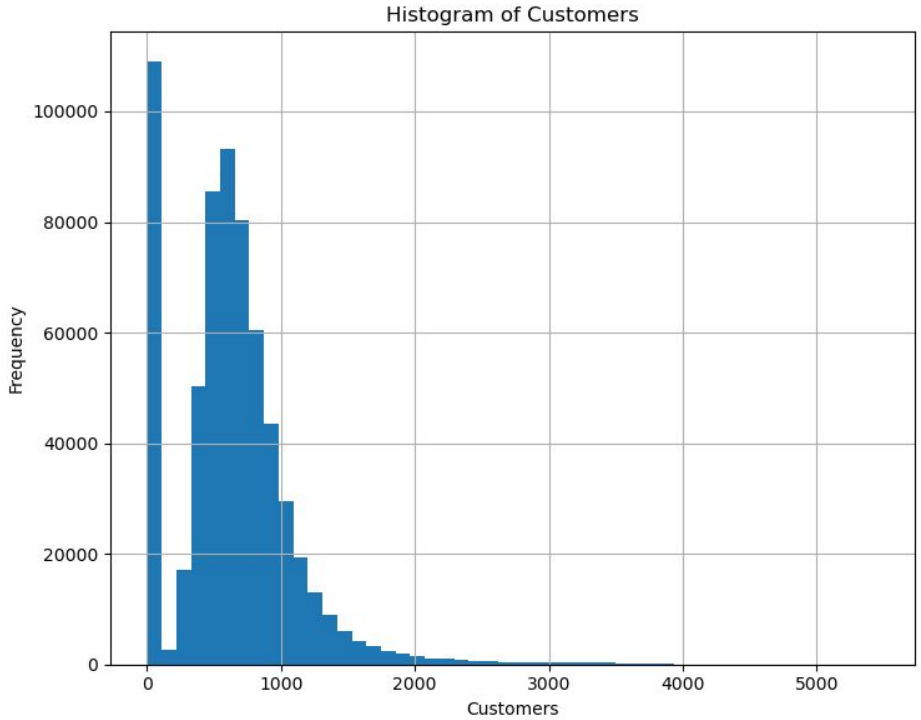
Data exploration



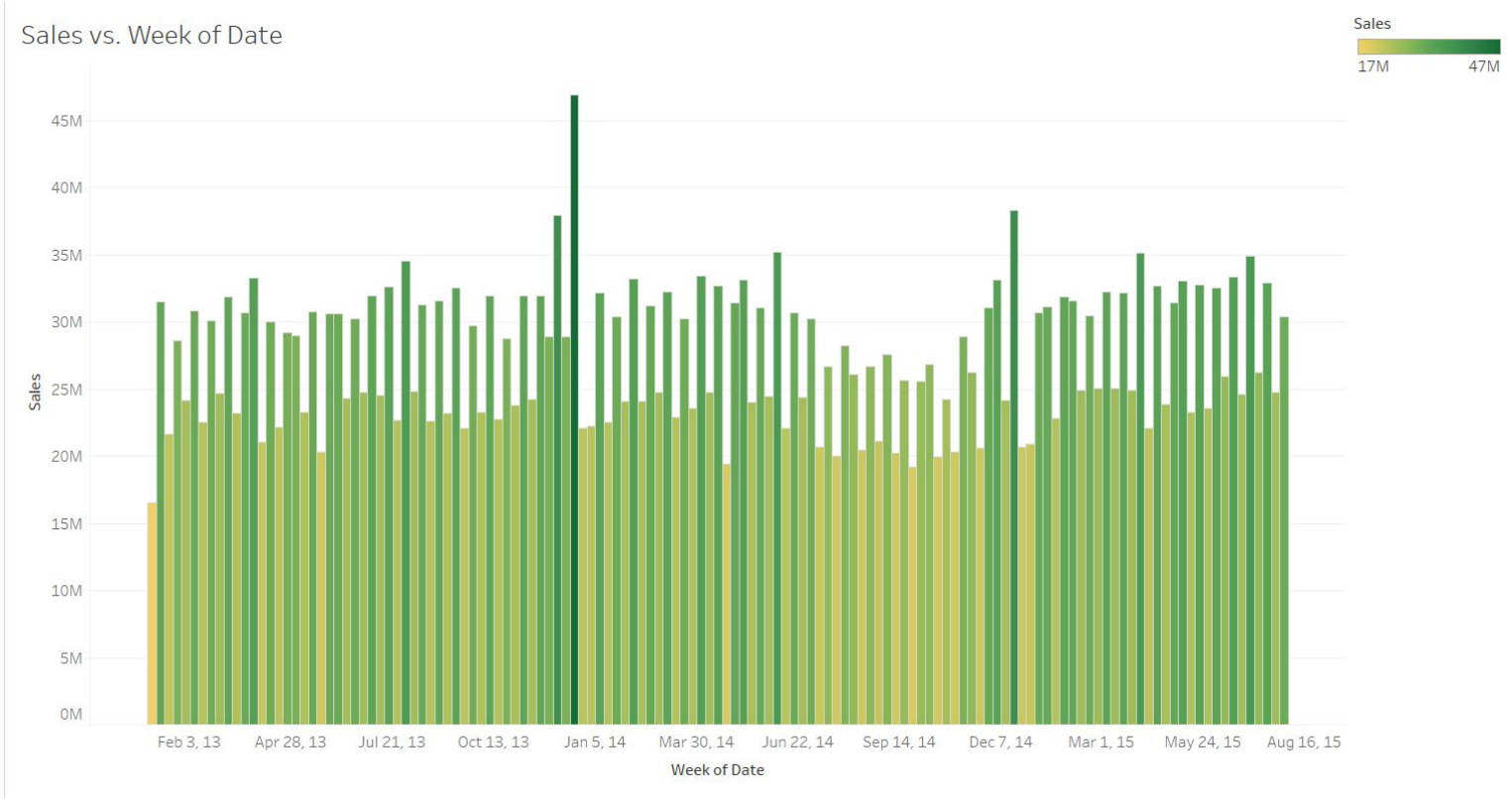


Data exploration

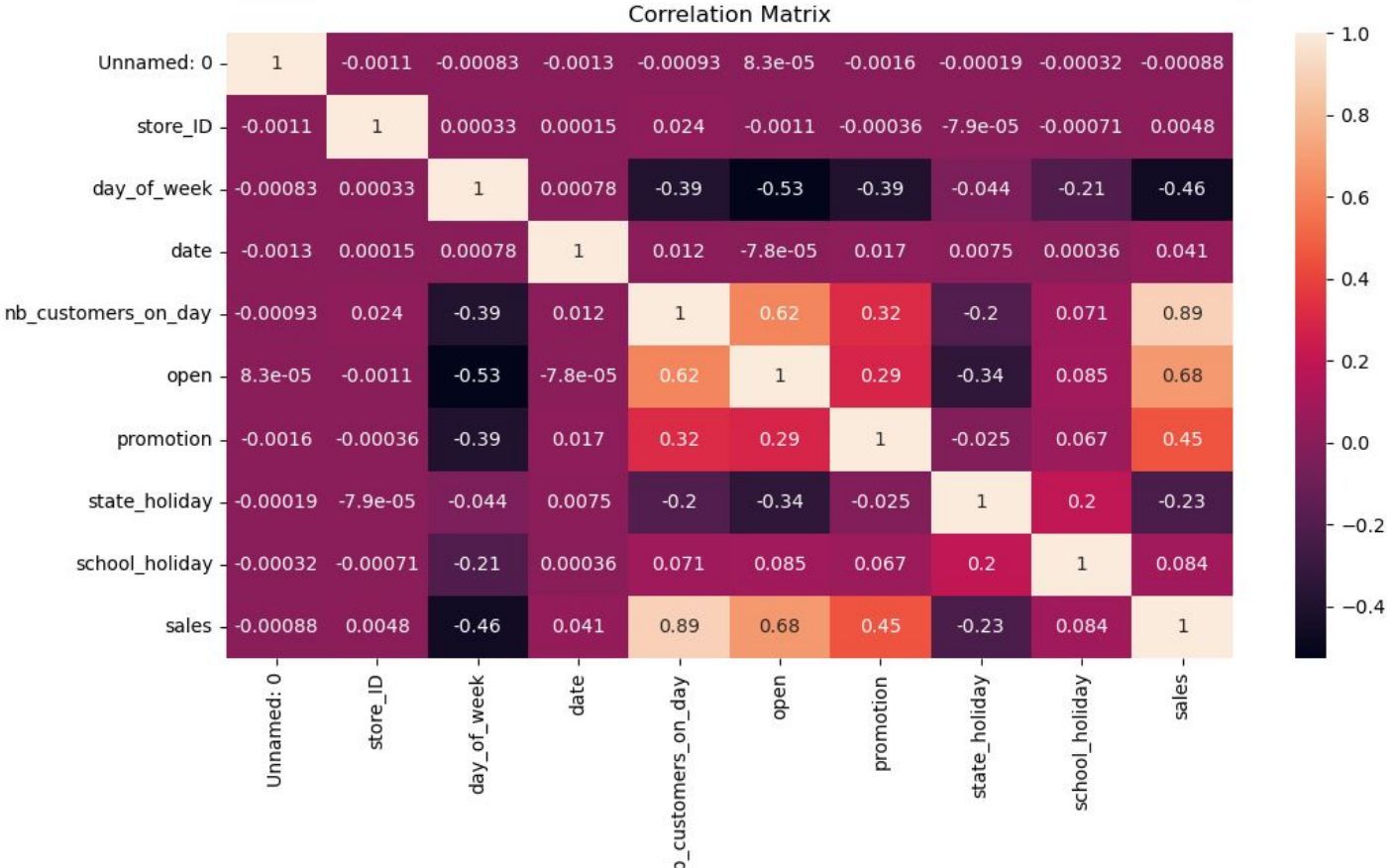




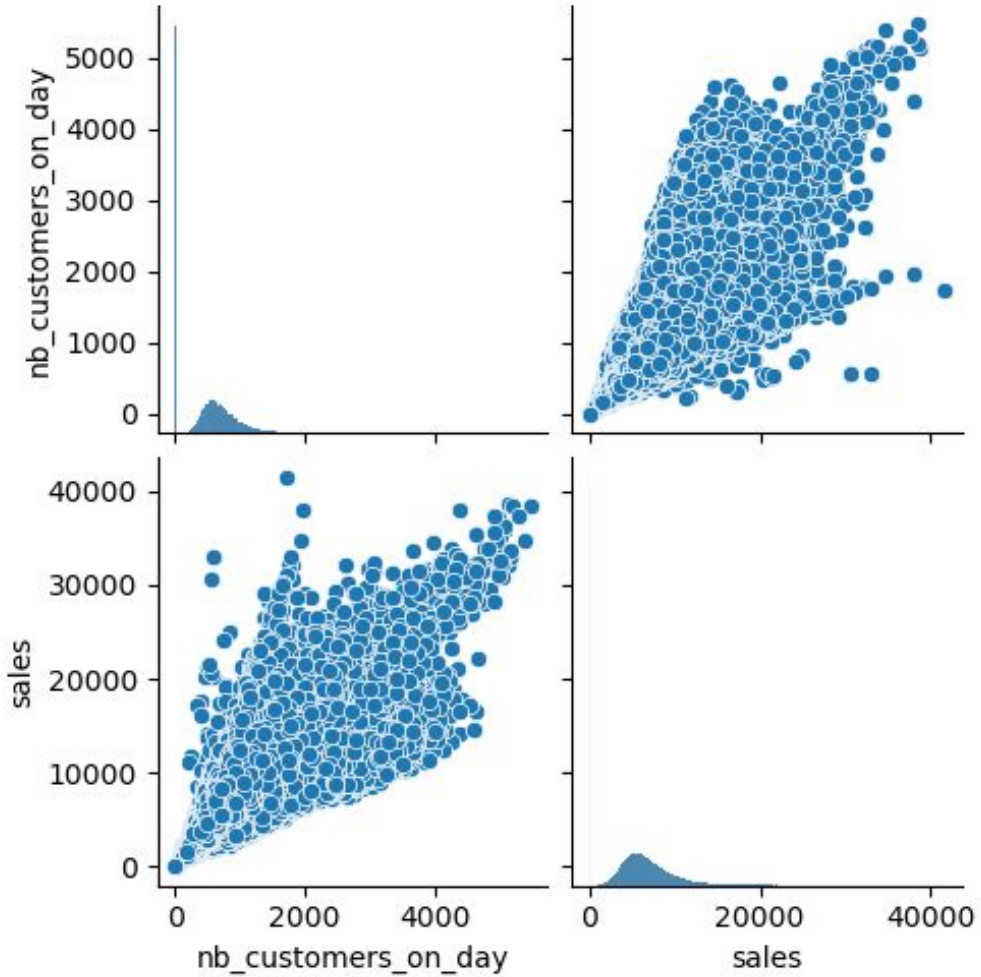
Data exploration



Data exploration



Data exploration



Models

Linear Regression

```
# create linear regression model
from sklearn.linear_model import LinearRegression

lr = LinearRegression()
lr.fit(X_train, y_train)
y_train_pred = lr.predict(X_train)
y_test_pred = lr.predict(X_test)

r2_train = r2_score(y_train, y_train_pred)
r2_test = r2_score(y_test, y_test_pred)

print(f"R2 score on training set: {r2_train:.2f}")
print(f"R2 score on testing set: {r2_test:.2f}")
```

```
R2 score on training set: 0.87
R2 score on testing set: 0.87
```

Decision Tree

```
# Desicion tree
from sklearn.tree import DecisionTreeRegressor

dt = DecisionTreeRegressor(max_depth=9)
dt.fit(X_train, y_train)

y_train_pred = dt.predict(X_train)
y_test_pred = dt.predict(X_test)

r2_train = r2_score(y_train, y_train_pred)
r2_test = r2_score(y_test, y_test_pred)

print(f"R2 score on training set: {r2_train:.2f}")
print(f"R2 score on testing set: {r2_test:.2f}")
```

```
R2 score on training set: 0.90
R2 score on testing set: 0.90
(Grid search)
```

Random Forest

```
rf = RandomForestRegressor(max_depth=9, n_estimators=200)
rf.fit(X_train, y_train)

y_train_pred = rf.predict(X_train)
y_test_pred = rf.predict(X_test)

r2_train = r2_score(y_train, y_train_pred)
r2_test = r2_score(y_test, y_test_pred)

print(f"R2 score on training set: {r2_train:.2f}")
print(f"R2 score on testing set: {r2_test:.2f}")
```

```
R2 score on training set: 0.90
R2 score on testing set: 0.90
(Grid search)
```

KNN ⭐

```
from sklearn.neighbors import KNeighborsRegressor
knn = KNeighborsRegressor(n_neighbors=9)
knn.fit(X_train, y_train)

y_train_pred = knn.predict(X_train)
y_test_pred = knn.predict(X_test)

r2_train = r2_score(y_train, y_train_pred)
r2_test = r2_score(y_test, y_test_pred)

print(f"R2 score on training set: {r2_train}")
print(f"R2 score on testing set: {r2_test}")
```

```
R2 score on training set: 0.923991977226772
R2 score on testing set: 0.903455527056366
(Grid Search)
```

SVR

```
svr = SVR()
svr.fit(X_sample, y_sample)
y_pred = svr.predict(X_sample)
r2 = r2_score(y_sample, y_pred)
print(f"R2 score is {r2}")
```

```
R2 score is 0.62
```

(Default rbf)

Neural Networks

```
# neural networks
model = Sequential()
model.add(Input(shape=(X_train.shape[1],)))
model.add(Dense(128, activation='relu'))
# model.add(Dropout(0.2))
model.add(Dense(64, activation='relu'))
# model.add(Dropout(0.2))
model.add(Dense(32, activation='relu'))
model.add(Dense(16, activation='relu'))
model.add(Dense(1, activation='linear'))

model.compile(optimizer=Adam(learning_rate=0.01), loss='mean_squared_error')
model.summary()

history = model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=12, batch_size=128,
verbose=1)
```

```
y_train_pred = model.predict(X_train)
y_test_pred = model.predict(X_test)

r2_train = r2_score(y_train, y_train_pred)
r2_test = r2_score(y_test, y_test_pred)

print(f"R2 score on training set: {r2_train:.2f}")
print(f"R2 score on testing set: {r2_test:.2f}")
```

```
R2 score on training set: 0.88
R2 score on testing set: 0.88
```

KNN

Highest R2 score on testing set (0.90)