Лабораторная работа № 8

Для выполнения лабораторной работы потребуется набор данных жилищного фонда Housing, содержащий информацию о зданиях в пригороде Бостона. Наблюдения представлены следующими характеристиками:

- **CRIM**: уровень преступности на душу населения по городу;
- **ZN**: доля жилых земельных участков, предназначенных для лотов свыше 25 000 кв. м футов;
- **INDUS**: доля акров нерозничного бизнеса в расчете на город;
- **CHAS**: фиктивная переменная реки Чарльз (= 1, если участок ограничивает реку; 0 в противном случае);
- **NOX**: концентрация окислов азота (частей на 10 млн);
- **RM**: среднее число комнат в жилом помещении;
- **AGE**: доля занимаемых владельцами единиц, построенных до 1940 г.;
- **DIS**: взвешенные расстояния до пяти Бостонских центров занятости;
- **RAD**: индекс доступности к радиальным шоссе;
- **TAX**: полная ставка налога на имущество на 10 тыс. долл.;
- **PTRATIO**: соотношение ученик-учитель по городу;
- **B:** вычисляется как *1000 (Bk – 0.63)^2*, где Bk – доля людей афроамериканского происхождения по городу;
- **LSTAT**: процент населения с более низким статусом;
- **MEDV**: медианная стоимость занимаемых владельцами домов в 1 тыс. долл.

В качестве целевой переменной будет использоваться переменная **MEDV**.

1. Загрузка данных:

```
from sklearn.datasets import load_boston
import pandas as pd

boston_dataset = load_boston()
boston = pd.DataFrame(boston_dataset.data, columns=boston_dataset.feature_names)
boston['MEDV'] = boston_dataset.target
boston.head()
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	18.7	396.90	5.33	36.2

2. Есть ли линейная связь между признаками и целевой переменной? Постройте корреляционную матрицу и оцените наличие линейной связи.

Корреляционная матрица – это квадратная матрица, которая содержит линейные коэффициенты корреляции, которые измеряют линейную зависимость между парами признаков.

3. Смоделируйте связь между ценами на жилье и **LSTAT** с помощью модели линейной регрессии, предварительно разделив данные на обучающие и тестовые.
4. Используя средневзвешенную квадратичную ошибку и коэффициент детерминации, оцените качество модели на обучающей и тестовой выборках. Какие выводы можно сделать?
5. Используя полиномы второй и третьей степени смоделируйте связь между **MDEV** и **LSTAT**. Оцените качество моделей с помощью коэффициента детерминации. Что вы можете сказать о полученных результатах?
6. Постройте регрессионную модель на основе деревьев решений. Оцените качество модели.
7. Постройте регрессионную модель SVM. Оцените качество модели.
8. Какую из моделей можно считать лучшей?
9. Улучшится ли качество модели, если использовать для обучения больше признаков? Какие признаки целесообразно добавить в модель?

Полезные советы

Вероятно, вам понадобится:

- Метод .corr() из библиотеки pandas

```
corr = boston.corr()
corr.style.background_gradient(cmap='coolwarm').set_precision(2)
```

- LinearRegression()
- Regression metrics

metrics.explained_variance_score(y_true, y_pred)	Explained variance regression score function
metrics.max_error(y_true, y_pred)	max_error metric calculates the maximum residual error.
metrics.mean_absolute_error(y_true, y_pred)	Mean absolute error regression loss
metrics.mean_squared_error(y_true, y_pred[, ...])	Mean squared error regression loss
metrics.mean_squared_log_error(y_true, y_pred)	Mean squared logarithmic error regression loss
metrics.median_absolute_error(y_true, y_pred)	Median absolute error regression loss
metrics.r2_score(y_true, y_pred[, ...])	R^2 (coefficient of determination) regression score function.
metrics.mean_poisson_deviance(y_true, y_pred)	Mean Poisson deviance regression loss.
metrics.mean_gamma_deviance(y_true, y_pred)	Mean Gamma deviance regression loss.
metrics.mean_tweedie_deviance(y_true, y_pred)	Mean Tweedie deviance regression loss.

- PolynomialFeatures()

```
X = boston[['LSTAT']].values
y = boston['MEDV'].values
regr = LinearRegression()

#Create polynomial features
quadratic = PolynomialFeatures(degree=2)
cubic = PolynomialFeatures(degree=3)
X_quad = quadratic.fit_transform(X)
X_cubic = cubic.fit_transform(X)

#Linear fit
X_fit = np.arange(X.min(),X.max(),1)[:,np.newaxis]
regr = regr.fit(X,y)
y_lin_fit = regr.predict(X_fit)
linear_r2 = r2_score(y,regr.predict(X))

#quadratic fit
regr = regr.fit(X_quad,y)
y_quad_fit = regr.predict(quadratic.fit_transform(X_fit))
quadratic_r2 = r2_score(y,regr.predict(X_quad))

#cubic fit
regr = regr.fit(X_cubic,y)
y_cubic_fit = regr.predict(cubic.fit_transform(X_fit))
cubic_r2 = r2_score(y,regr.predict(X_cubic))

#plot results
plt.scatter(X,y,label='training points',color='lightgray')
plt.plot(X_fit,y_lin_fit,label='linear (d=1), $R^2=%.2f$' % linear_r2, color='blue', lw=2, linestyle=':')
plt.plot(X_fit,y_quad_fit,label='quadratic (d=2), $R^2=%.2f$' % quadratic_r2, color='red', lw=2, linestyle='-')
plt.plot(X_fit,y_cubic_fit,label='cubic (d=3), $R^2=%.2f$' % cubic_r2, color='green', lw=2, linestyle='--')
plt.xlabel('% lower status of the population [LSTAT]')
plt.ylabel('Price in $1000\'s [MEDV]')
plt.legend(loc='upper right')
plt.show()
```

- DecisionTreeRegressor()
- SVR()